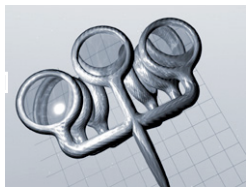
Deliverable D5.1: Report on method and language for the production of the augmented document representations

Erick Alphonse, Sophie Aubin, Julien Derivière, Thierry Hamon, Dunja Mladenic, Adeline Nazarenko, Claire Nédellec, Thierry Poibeau, Davy Weissenbacher, Qiang Zhou



002068

ALVIS

Superpeer Semantic Search Engine

STREP / IST

Deliverable D5.1

Report on method and language for the production of the augmented document representations

Title of Contract	**ALVIS - Superpeer Semantic Search Engine**
Acronym	**ALVIS**
Contract Number	**IST-1-002068-STP**
Start date of the project	**1.1.2004**
Duration	**36 months, until 31.12.2006**
Document name	**Deliverable D5.1 – Report on method and language for the production of the augmented document representations**
Date of preparation	**December 31, 2004**
Author(s)	**Erick Alphonse, Sophie Aubin, Julien Derivière, Thierry Hamon, Dunja Mladenic, Adeline Nazarenko, Claire Nédellec, Thierry Poibeau, Davy Weissenbacher, Qiang Zhou**
Coordinator of the deliverable	**Adeline Nazarenko** **Phone: (33) 1 49 40 40 89** **Fax: (33) 1 48 26 07 12** **Email: nazarenko@lipn.univ-paris13.fr**
Document location	**http://project.alvis.info/copies/2005_06/ALVIS_D5_1_20043112_P13_AN.pdf**

Project funded by the European Community under the
"Information Society Technology" Programme (2002-2006)

Abstract

This deliverable describes the specifications of the Natural Language Processing line to be developed within the Work Package 5 (WP5) in the Alvis Project.

The WP5 is in charge of the NLP aspects of the Information Retrieval process. Its main objective is to enrich and normalise the crawled documents provided by the WP7 prior to their indexing. In this respect, WP5 output is a set of annotated documents, which are provided either to WP2 as input of its probabilistic model, which in turn delivers document to WP3 for indexing, or to WP6 as acquisition corpora.

This deliverable presents the objectives of the WP5 process for document annotation. It introduces the basic notions of linguistics that are used in the following. The core section presents the various types of annotations that WP5 is expected to produce and describes the format in which these annotations are encoded.

In addition to that, this deliverable also gives an overview of the WP5 processing line, its architecture for the various languages studied in Alvis and the various NLP components that produce annotated documents.

Keywords	Natural Language processing, linguistic annotations, stand-off annotations, XML
Work package	WP5
Deliverable type	article

Change log	
6.1.2005	Initial version.
17.1.2005	Minor changes.

Executive summary

This deliverable describes the specifications of the Natural Language Processing (NLP) line to be developed within the Work Package 5 (WP5) in the Alvis Project.

WP5 is in charge of the NLP aspects of the Information Retrieval (IR) process. Its main objective is to enrich and normalise the crawled documents provided by the WP7 prior to their indexing. In this respect, WP5 output is a set of annotated documents that are provided either to WP2 as input of its probabilistic model, which in turn delivers document to WP3 for indexing, as a production flow of documents. WP5 also aims at enriching acquisition corpora for domain specific resource acquisition by WP6, as acquisition flow of documents.

This deliverable presents the objectives of the WP5 process for document annotation and gives an overview of the WP5 document processing line.

The WP5 NLP line aims at enriching documents with annotations at the morphologic, syntactic and semantic levels for four different languages (English, French, Slovene and Chinese). Depending on the resources available, the quality of the NLP line in the different languages and the requirements of the other WPs, various levels of annotations can be produced. The integration of specialized resources (specific knowledge in the form of dictionaries, terminologies and ontologies, acquired in WP6) should enhance the standard NLP line by tuning it for specific domains.

The result of NLP analysis is attached to the document in the form of XML annotations which are not traditional inserted mark-up but to stand-off annotations, so as to separate the annotations themselves and the text of the document to annotate, which thus remains uncorrupted.

The architecture of the WP5 NLP line is developed as a generic one, in which various NLP modules can be integrated. The main modules handle the recognition of named entity and terms, the morphological reductions and the tagging of semantic types and relations. Syntactic parsing is crucial from an acquisition point of view.

Contents

1 Introduction

This deliverable describes the specifications of the NLP line to be developed within WP5 in the Alvis Project.

The WP5 is in charge of the NLP aspects of IR process. Its main objective is to enrich and normalise the crawled documents provided by the WP7 prior to indexing. In this respect, WP5 output is a set of annotated documents that are provided either to WP2 as input of the document probabilistic model computing or to WP6, as acquisition corpora.

The section 2 presents the objectives of the WP5 process for document annotation. The section 2.4 introduces the basic notions of linguistics that are used in the following. The section 4 is the core section: it presents the various types of annotations that WP5 is expected to produce and it describes the format in which these annotations are encoded. The full XML DTD is presented in an annex (section 9).

The two remaining sections give an overview of the WP5 processing line, its architecture for the various languages studied in Alvis (section 5) and the list of the various NLP components that produce annotated documents (section 6).

2 Production of annotated documents

The goal of the NLP line of WP5 is to produce documents enriched with linguistic annotations (annotated documents). The type and quality of the annotations vary with to the following factors

- The way the annotated documents are used: they can be used either for IR or for acquisition and, in both cases, with various levels of annotations (see section 2.1);
- The availability of domain-specific resources: the more domain specific knowledge is available, the richer the document annotations can be. In this respect, WP5 and WP6 are strongly connected (see deliverable 6.2 for more details on knowledge acquisition);
- The language of the documents: the intrinsic complexity and the state of the art in NLP differs from one language to another (see section 2.2);
- The volume of textual data to process: since NLP is known as expensive computing, the deepness of document analysis will depend on the efficiency of WP5 analysis and the volume of documents to be analysed.

One of the objectives of the Alvis project is to test the various combinations of annotations to identify which ones have a significant impact on IR results.

The present deliverable describes the various types of annotations that can be produce by WP5. From this starting point, the state of the art on the integration of NLP in IR and the behaviour of the other Alvis modules (mainly the probabilistic model), the Alvis partners will have to decide which specific combinations of annotations should actually be tested in Alvis (see the forthcoming deliverable 5.2).

2.1 Production vs. acquisition

WP5 produces annotated documents to enhance the performances of IR. The documents output by WP5 are then indexed to be included in the search engine document base. In this respect the main flow of documents, the **production flow**, is the following: the documents that are crawled (WP7) are annotated (normalisation and/or enrichment) by WP5 and then indexed by WP3 according to a specific probabilistic model designed in WP2.

However, the document annotation is very often domain dependant and WP5 makes use of domain specific resources to produce documents with domain specific annotations. WP5 therefore has to handle a second flow of documents (**acquisition flow**) for the acquisition of these specific resources to be used on the production flow. This second acquisition flow is the following: a given set of documents (an acquisition corpus) is processed by WP5 and delivered with the required level of annotations to WP6. Specific resources are then acquired in WP6, thanks to the pre-processing of the acquisition corpora in WP5.

Both flows of annotated documents are produced using the same NLP components. This point is important to ensure that the specific resources acquired are homogeneous to the document that must be annotated in the production flow.

The two production and acquisition flows differ in volume of textual data. In the production flow, WP5 has to annotate the document base that must be indexed and search for in IR. In the acquisition flow, WP5 has to process only a small amount of representative data necessary to acquire the needed resources..

2.2 Multilinguism

In Alvis, "multilinguism" means that the document normalisation process is performed in different languages so that various search engines can be developed for various languages. The trans-language problem (a query in a source language is matched on documents written in a different target language) is not addressed here.

Four languages are addressed in Alvis: English, French, Slovene and Chinese. However, the linguistic characteristics, the state of the art in NLP, the requirements for IR and therefore the level of achievement differ from one language to another. In the following, English is considered as the basic language for which the full NLP line is developed. The three other languages are presented by comparison with the English one.

2.2.1 English

Natural language raises several problems that affect the performance of Information Retrieval (IR) in terms of precision, recall or efficiency:

- Polysemy lowers the IR precision
- Paraphrasing lowers the IR recall
- The vocabulary structure strongly increases the IR complexity

Polysemy

The fact that a given word or expression A may have several meanings (S1 and S2) affects the precision in IR results. Considering a request with A as a key word. If the user has the meaning S1 in mind, only the documents with A meaning S1 are relevant for him/her whereas a basic word based search engine will retrieve all the documents where A occurs (either with the meaning S1 or with the meaning S2) and part of them are irrelevant.

Under the term "Polysemy", we actually refer to various linguistic phenomena ranging from homography (a same character string belongs to two different words, such as *suis* in French which is a verbal form belonging either to the verb *être* (*to be*) or to the verb *suivre* (*to follow*)), to very fine-grained sense distinctions (such as *bank* for which 10 different senses are listed in WordNet [1]).

The polysemy must be handled by disambiguation techniques both in the query and in the document collection,.

The present deliverable deals with the document processing. Due to the heterogeneity of the polysemous phenomena, reducing polysemy require various techniques of word sense disambiguation based on syntax, context and/or background knowledge. One of the difficulties in handling polysemy is that the sense distinctions are relative rather than absolute.

Paraphrasing

As opposed to polysemy, which affects precision, the synonymy and more generally the existence of paraphrases affect the recall in IR results.

We consider that two expressions are the paraphrase of each other if they are different wordings for the same meaning (although meaning similarity is a relative and contextual notion). This includes synonymy, which corresponds to lexical paraphrases (two words with the same meaning, such as *color*/*colour* or *increase*/*raise*), but covers many other phenomena (*composition sanguine/composition du sang*, *blood composition/composition of the blood*, *X is expressed by Y/expression of Y by X,* etc.).

The existence of paraphrases affects the recall in the IR results, since the user does not usually think of the various equivalent A' of the key word A that he/she looks for. The search engine therefore misses the various documents

[1] http://www. cogsci.princeton.edu/~wn

containing the words or expressions A' that have more or less the same meaning than A but that correspond to different character strings in the documents.

Handling the paraphrases can be handled both on the query side (query expansion) and on the document side by normalising the documents prior to their indexing. In this deliverable, we focus on this later point.

Vocabulary structure

The vocabulary structure is common to most languages and most documents or corpora. The size V of the vocabulary of any corpus is always smaller than the size T of the text as some items occur several times. However, several characteristics must be underlined:

- V increases as T increases;
- the proportion of hapax (vocabulary items that occur only once in a given corpus) is always important and it increases with T;
- the most frequent items are more or less the same for all the corpora in a given language. For English and French, they are the grammatical words.

These characteristics explain that the presence matrix of the vocabulary items of a corpus on the various documents that compose the corpus is very sparse, which has strong implications for IR. One of the goals of the document indexing process is to reduce the size of V by comparison with T while keeping the items that are the most significant on semantic grounds.

In Alvis, the objectives of the English NLP is to develop a document normalisation line that enables to test various NLP pre-processing steps and their impact on IR performances.

2.2.2 French

French language is very similar to the English one except for:

- Alphabetical characters, since the French language has accented vowels;
- The morphology, which is must richer in French than in English. See below, section 3.2.

The NL processing of French and English are therefore very close except for the morphological analysis, which is more complex. Thus, the traditional algorithm for morphological analysis of English (mainly stemming) does not apply for French.

The state of the art in NLP is also similar for English and French except that available resources (either knowledge bases ad processing tools) are not as numerous and rich as for English.

The development of the French NLP line in Alvis, although not complete (e.g. the syntactic analysis may not be tested for French), will enable to test the robustness of the WP5 architecture.

2.2.3 Slovene

The same difficulty arises for the Slovene language, at a higher degree. Slovene is a highly inflected natural language, having up to 30 different word forms for the same normalized word, carrying almost the same meaning. In the context of IR, this is a major source of paraphrases and their reduction is a necessary. This is the role of the morphological analysis of Slovene (especially lemmatization, see below, section 3.2).

A lot of work has been carried out in word normalisation, particularly for English, *e.g.* the well-known Porter stemmer •[7] , but except for research prototype •[3] •[4] , no algorithm is readily available for lemmatisation of Slovene words.

In Alvis, for the Slovene, the focus is therefore on the morphological analysis.

2.2.4 Chinese

The Chinese language is different from many European languages, such as English, French and Slovene. The Chinese text is a sequence of Chinese characters. There is no separation mark between different words. So the word breaking is the first task for Chinese NLP. Another characteristic for the Chinese language is that there are not any special marks

for the Chinese proper names, such as the capitalized marks for the English proper names, so the proper name identification is also a difficult problem in Chinese NLP.

Nowadays, the performance of automatic segmentation, POS tagging, name entity identification and chunking in the Chinese language are very good. But the performance of Chinese syntactic parser is still worse than English parser. And the automatic semantic analysis in the Chinese language is in its primitive stage. So in Alvis project, we will focus on the segmentation, POS tagging and chunking processing for the Chinese language.

2.3 OpenSource question

What is planned in Alvis is to produce an OpenSource NLP line with plug-in components that are either available (we choose OpenSource ones whenever possible) or developed within Alvis (OpenSource code).

It must be noted however, that many NLP components are not available as OpenSource ones. Very often, either the source code or the knowledge base is not publicly available. Even if the NLP state of the art for French is quite similar to that of English, in the OpenSource perspective, the situation is much worse.

2.4 Various levels of annotation

WP5 provides other WPs with annotated documents. The level of annotation of these documents depends on:

- The quality of NLP that WP5 is able to produce. This quality itself depends on:
  - The types of NLP components available (which vary from one language to another),
  - The reliability of their results,
  - The resources that are provided as inputs to these components to customize their behaviour;
- The type, volume, quality and depth of document analysis that other WPs (mainly WP2, WP3 and WP6) require from WP5. These requirements depend on the types of knowledge WP6 has to acquire on one hand and the type of NL information exploited in IR experiments done by WP8, on the other hand.

The present document presents the various types of annotation that WP5 will be able to produce, with some indications on the expectable quality of the results.

3 Basic notions

This section describes the various levels of linguistic information that are relevant for NLP or produced by WP5 processing line in the form document annotations.

It defines the terminology that is used henceforth.

3.1 Document structure

a. Definition

The document structure describes the sectioning of the document. It provides information about the logical organisation of the document according to a hierarchical structure: section, subsection, paragraph, and list. Structural information could also identify the structural function of each textual segment of the document: title, authors, date of publication and hypertext links.

Except for specific tasks such as text summarisation and document fine-grained indexing, the use of structural information is very limited: NLP tools usually exploit the sentence as basic textual unit.

b. How is document structure handled in WP5?

Sentence segmentation is a crucial step in WP5, since it affects almost all NL processes.

Basically, strong punctuations (. ! ?) are used to identify the end of the sentences. The various parts of the documents, such as titles or list items, are considered as sentences even if they do not end with such marks. In that respect, structural information helps the sentence segmentation.

The annotation of the document structure, except for the identification of sentences, is out of the scope of WP5. The analysis of the structure of documents is processed prior to WP5 in WP7. The canonical document is provided with basic structural information: section, list and its items, and hypertext links. In WP5, the boundaries of various parts of the document (section, title, list items) are considered as strong punctuation marks.

For instance, the document is provided as described in the example below.

Example of a structured document

```
<canonicalDocument>
<section>
Carnivorous Plants
    Carnivory is a useful strategy for plants growing in nutrient poor environments.
Throughout the world there are over 500 species of carnivorous plants in many different
families.
In North America carnivorous plants are mostly found in boggy environments - they include
the bladderwarts, venus' flytrap, the pitcher-plants and the sundews. Each of these
plants has its own distinctive way for trapping prey.
</section>
</canonicalDocument>
```

It is translated in the following format for the linguistic processes (each section is represented on one line).

Example

```
Carnivorous Plants
Carnivory is a useful strategy for plants growing in nutrient poor environments.
Throughout the world there are over 500 species of carnivorous plants in many different
families.

In North America carnivorous plants are mostly found in boggy environments - they include
the bladder warts, venus' flytrap, the pitcher-plants and the sundews. Each of these
plants has its own distinctive way for trapping prey.
```

3.2 Morphology

a. Definition

The morphology analysis focuses on the form of textual units, usually referred as words, even if there is no consensual definition of what a "word" is.

The morphological analysis includes:

- The identification of the textual units, i.e. the segmentation of the character string into a sequence of words. The segmentation quality ranges from a character-based segmentation (each alphabetical character string separated by blank and punctuation characters is considered as a word) to a more linguistically based segmentation. In this latter case, some polylexical units can be considered as words (*Bacillus Subtillis* in Biology; *did make* in English; *pomme de terre*, *chou-fleur, ne…pas* in French) and a single character string can be viewed as the concatenation of several words (*des="de les"*, in French).
- The analysis of the word internal structure, which can be decomposed into two different questions:
    - The flexional structure: the word is considered as a lemma (the word canonical form) combined with flexional affixes (mainly suffixes): *derived = deriv(e)+ed* (English); *chanteront = chant(er) + eront* (French). The affixes usually bear morphological features such as tense, person and number. The importance and the complexity of flexional analysis vary from one language to another. In French, on

average, there are 7 different forms attached to a given lemma (due to the rich French conjugation system); Slovene has an even richer system with 6 different noun cases for instance.

- The derivational structure: the word is considered as a stem (or root form) combined with one or several derivational affixes (mainly prefixes and suffixes): derivation = deriv(e) + ation (English); contradiction = contra + dict + ion (French). Two different forms derived from the same stem are usually considered as different words in traditional lexicography.

b. How morphology is handled in WP5 ?

Various modules are involved in the morphological analysis:

The **segmentation** is processed in several steps:

- The tokenization first splits the text character string into a sequence of tokens: either alphabetic or numeric tokens, punctuation tokens and separator tokens.
- The segmentation in words is processed afterwards. It identifies the word boundaries in terms of the tokens that have been previously identified. The tokenization and the word segmentation may diverge: a word such as *pomme de terre* in French is made of several tokens whereas a single token like *des* corresponds to two different words. In English the sequence of 3 tokens *don't* is considered as a sequence of two words (*do*+*not*).
- The processes of Name Entity and term recognition may also contribute to linguistic segmentation, since they identify textual fragments that can be considered as single units from a referential or terminological point of view (*Mrs Antonella Biretti*, *heart attack, sigma K* in biology) (see below section a.).

S**temming** aims at identifying the stem from which a word *w* derives, or more pragmatically the family of words to which *w* belongs. Stemming can thus be considered as a word clustering process. In Alvis, an approximate stemming (as opposed to a linguistically grounded one) may be performed, if it is relevant for the probabilistic analysis and indexing of documents. Various algorithms have been proposed to handle stemming as suffix removing (for instance Porter or Krovetz stemming algorithms •[7] •[5]).

L**emmatization** aims at identifying the canonical form of a word (infinitive form for verbs, singular form for nouns, pronouns…).

The lemmatization is the process of finding the normalized form (lemma) of a word. It is similar to word stemming but it does not require to produce a stem of the word but to replace the suffix of a word with a (typically) different word suffix to get the normalized word form. For instance, the suffixes of words "working", "works", "worked" would change to get the normalized form "work" standing for the infinitive; in this case, both the normalized word form and the word stem are equal. Sometimes the normalized form may be different than the stem of the word. For example, the words "computes," "computing", "computed" would be stemmed to "comput", but their normalized form is the infinitive of the verb "compute".

M**orphological tagging** associates various morphological features to a given word. Traditional features deals with tense, number, gender and case. Their relevancy varies from one language to another.

3.3 Syntax

a. Definition

The syntactic analysis deals with the grouping of words that form the sentence structure (these groupings are called "phrases"). It aims at identifying the syntactic categories of words and phrases as well as the relationships that hold between the various words or phrases of a given sentence. The syntactic analysis therefore performs two different tasks:

- The tagging consists in assigning a syntactic category (often part-of-speech categories) to a word (Noun, Verb, Determiner, etc.) or a phrase (Noun Phrase, Verb Phrase, etc.). For computational reasons, the tagging of words (traditionally referred as *tagging*) is usually performed independently and prior to the parsing, while the parser performs the tagging of the phrases.
- The parsing aims at identifying the phrases that compose the sentence. Depending on the parser formalism, the parsing also aims at identifying the role of each word or phrase within the sentence and with respect to other words or phrases.

b. How syntax is handled in WP5

In Alvis, two separate modules, themorpho-syntactic tagger and the parser, perform the syntactic analysis.

The **morpho-syntactic tagger** associates various morphological (see above) and syntactical features (usually part-of-speech) to the words that form a sentence. A tagger assigns a syntactic category to each word on the basis of a dictionary and of context rules for ambiguous words (e.g. "show" is either a noun or a verb) or unknown words that do not appear in the dictionary.

The **syntactic parser** identifies the phrases and relations. In Alvis, in the perspective of ontology learning that requires syntactic relation parsing, syntactic dependency analysis is preferred to constituency analysis. The syntactic parser creates links between pairs of words and types these links as "subject-verb", "noun-adjective", "adverb-verb", "verb-object", etc. The conjunction of all the links gives the parse of the entire sentence.

3.4 Semantics

a. Definition

The semantic analysis aims at building a formal representation of a sentence or text meaning. Various formalisms have been studied in the past to encode such a semantic representation, ranging from predicate calculus, conceptual graphs, discourse representation structures, etc. In Alvis, the semantic analysis is restricted to

- the identification of the semantic textual units (name entities and terms, mostly) that refer to the relevant domain objects. Note that identifying those semantically relevant units may also contributes to syntactic segmentation;
- their semantic typing,
- the tagging of domain specific relations between them.

Named entities

Named entities refer to meaningful linguistic units of a text, for example person names (*Jacques Chirac*, *George W. Bush…*), location names (*Washington*, *London*), dates (*The 3rd of July, 2000*), etc. These units can be different according to the domain. For example, gene names are named entities in the biological domain. They are also distinct from technical terms (non referential linguistic units), even if the limit is not always so clear.

Named entities are especially relevant for text indexing. For example recognizing person names often increase the relevance rate of retrieval engine in companies or scientific survey offices.

The identification of named entities consists in identifying the nouns or noun phrases that refer to specific and well-identified domain entities (mostly proper nouns). It is an important task in natural language processing, because this type of expression plays an important role for the referential anchoring of many corpora (newspapers, corporate documents, e-mails...). It is therefore important to be able to identify these expressions either for specific applications (e.g. to index documents by proper names lists) or for general research purposes (e.g. to improve the syntactic analysis of a text). Many research projects have addressed the issue of proper names identification in newspaper texts; in particular, the Message Understanding Conferences (MUC). In these conferences, the first task to achieve is to identify named entities, i.e. proper names and also temporal and numerical expressions. This task is generally viewed as being generic, in the sense that all texts use such expressions and their identification seems a priori independent of the discourse domain or textual genre.

Terms

A terminology is the main knowledge source about the technical vocabulary of a specialized domain. First of all, it lists the terms (simple or complex lexical units) of this domain. The terms point out specific concepts and are acknowledged as such by a large majority of the experts of that domain.

Identifying terms in corpora is important since terms refer to the domain concepts and are the textual units, either words or phrases, that are relevant from a semantic point of view. Terms can be automatically extracted or provided by resources as thesaurus, specialized dictionaries, and glossaries. One difficulty of terminology analysis comes from the

fact that terms do not always occur in the same form. There often occur under a morpho-syntactic or semantic variants form :

- Morpho-syntactic variation records the inflectional, derivational variants (*Aortic stenosis* / *Stenosis of the aorta*) but also permutation (*Aortic Subvalvular Stenosis* / *Subvalvular Aortic Stenosis*) or coordination of terms (*Mitral Stenosis* / *Aortic and Mitral Stenosis*).
- Semantic relations describe the hierarchical structure of the vocabulary (*Aortic Valve Insufficiency* Is_a *Heart Valve Diseases*), or similarity between words (*Aortic Incompetence* is_synonymous_of *Aortic Valve Insufficiency*). This latter type of information will not be exploited as such in Alvis.

Semantic typing

The semantic typing consists in attaching concept names, or categories to the textual units (usually named entities and terms). Concepts may refer to various levels of generality. In the ontology, the generality relation between concepts is expressed in the form of multiple hierarchies and the attachments to terms of the domain are represented in a formal and declarative way.

Semantic relations

Two types of semantic relations are considered:

- The domain specific relations (e.g. X is a symptom of Y) are domain knowledge that help to interpret the document content. Making explicit these relations enriches the semantic content of the document: whether it has an impact on IR performances is an open question in Alvis.
- The anaphoric relations link two textual units, the anaphoric expression (or anaphor) and the antecedent. The antecedent provides the information necessary to interpret the anaphor. The anaphoric expression usually refers "back" to a previous textual unit (*anaphora*, strictly speaking) but it can also refer backward to a definite expression (*cataphora*). Here, we use the term "anaphora" in its usual meaning to refer to both phenomena.

b. How is semantic analysis handled in WP5?

The quality of that semantic analysis depends on the IR task and the resources available. Several modules perform the semantic analysis in Alvis.

The **name entity tagging** and the **terminology tagging** are performed independently to each other although one may consider that named entities are nothing else than specific terms. Since the special forms of named entities is a handicap for morphological and syntactic analysis, it is important that named entities are identified prior to any other processing.

Depending on the IR task, the NLP line runs on different modes that affect the **semantic category and relation tagging** strategy:

- If the document collection is parsed by a syntactic parser, the semantic analysis can exploit this information to achieve a good tagging precision:
    - Semantic typing is constrained by syntactic conditions for *Word Sense Disambiguation* (WSD), in a similar way as selectional restrictions. It is particularly useful in biology where the homonyms are very frequent.
    - The tagging of domain specific relations is done with the help of sophisticated patterns, the conditions of which include syntactic as well as semantic criteria. In this mode, the tagged relations are explicitly expressed in the text.
- In the case where the document is not syntactically parsed.
    - Semantic typing is done on the only basis of the object neighbourhood in the document with a lack of precision and a gain in computational time.
    - The documents are enriched with *all* domain specific relations that are represented in the ontology between the entities that are mentioned in the document, even if the relations themselves are not explicitly mentioned in the document

Notice that this latter mode can also be applied on a parsed document collection. Such a semantic typing would be useful in the case where the ontology or part of it does not include syntactic disambiguation conditions. However, one may prefer the "light" mode for specific domain relations even for parsed documents, depending on the IR task.

In the following, for the sake of clarity, we will annotate the examples according to the first strategy.

There is finally a separate **anaphora resolution** module that identifies the antecedents of the anaphoric pronouns of a given document.

4 Various types of linguistic annotations

This section justifies the choice of stand-off annotations rather than inserted mark-up (section 4.1) and lists the various types of annotation that may be produced by WP5 NL processing lines (sections 4.2 to 4.4). The complete DTD for linguistic annotations is given in annex (see page 56). Chinese annotations differ significantly from the annotations for other languages. They are presented apart in section 4.5.

Most examples are English ones for sake of simplicity, except for other language specificities, whenever necessary.

4.1 Annotation principle

The Alvis principle for the linguistic part of document annotation (see the overall metadata format for enriched documents, task 3.2) is based on the study of existing standard (section 4.1.1). We made the choice of stand-off annotations rather than inserted mark-up (section 4.1.2). These stand-off linguistic annotations are encoded as XML elements and form the linguistic subsection of the document overall annotation format.

4.1.1 Linguistic annotation standards

The problem of representing linguistic annotations is not new. It has been widely studied since the beginning of the nineties and several ad hoc formats have been proposed (see •[1] •[4]). The efforts to unify these formats in order to allow interoperability among NLP tools are recent. The two following projects are going to merge:

- The TEI: an international and interdisciplinary standard that helps people to represent all kinds of literary and linguistic texts for online research and teaching, using an encoding scheme that is maximally expressive and minimally obsolescent;
- An ISO proposition (TC37SC4/TEI) currently under definition, which will include a Feature Structure Representation, a Morpho-Syntactic Annotation Framework, a Category Data Repository, a Linguistic Annotation Framework, a Lexical Mark-up Framework and some Data Category S-Electronic Lexical Resources.

The Alvis proposition is based on these standards but:

- The current ISO drafts present propositions that are far from definitive and which may evolve in a significantly different way;
- Our goal is not as general as that of the TC37SC4/TEI: strictly complying with the norm would make our annotation formalism more complex whereas a light version is sufficient for Alvis needs.

On the whole, we follow the TC37SC4 principles without sticking to the current version. We adopt the principle of stand-of annotation and we distinguish several types of annotations (e.g., tokens, words, phrases, sentences, syntactic and semantic relations), see sections 4.2 to 4.4 below for more details. Each annotation being provided with references to some other annotations when needed.

4.1.2 Arguments towards stand-off annotations

The principle of stand-off annotations is to separate the text of the document to annotate and the annotations themselves, as opposed to traditional inserted annotations whose mark-up is inserted in the text that is annotated.

They are many arguments towards such stand-off annotations:

- The initial textual data may be read-only and/or very large, so copying it to insert mark-up may be unacceptable;

- The distribution of the initial data may be controlled whereas the mark-up is intended to be freely available;
- Since the stand-off annotations do not pollute the initial textual data, it is always possible to refer to it;
- Stand-off annotations allow embedded annotations that are impossible for inserted mark-up. It is therefore easier to encode:
    - Concurrent annotations produced by different NLP tools;
    - Non linear elements, which may be relevant linguistic entities such as “to… decide” in “to completely decide” of “ne… pas” in “je ne mange pas” (“ne… pas” is the traditional negative adverb in French language);
    - Relations (grammatical functions, semantic relations) between elements belonging to various levels in the hierarchy of annotations;
    - Complex annotations overlapping two different structures, such as anaphora relations that may rely two elements belonging to two separate sentences;
- New levels of annotations can be added without disturbing the existing ones;
- Editing one level of annotation has minimal knock-on effects on others;
- Each level of annotation can be stored and handled separately, eventually in several files.

The main drawback of the standoff annotation principle is that it is difficult and computationally expensive to rebuild the textual signal from the list of annotations.

4.2 Textual units

Different levels of textual units are relevant for NLP. In Alvis, we take five levels into consideration. At a basic level, the text is segmented into tokens; other levels are built on the token level and are linguistically grounded: we distinguish the words, the phrases, the semantic units and the sentences.

The properties of these textual segments are described in separated levels of annotations (see sections 4.3 and 4.4 for morpho-syntactic and semantic annotations respectively):

- The tokens correspond to the basic segmentation level;
- The words are made of tokens;
- The phrases are made of words and/or other phrases;
- The semantic units are made of phrases and/or other semantic units.

The Alvis format for these textual units is homogeneous from one level to another. Except for tokens, each textual unit has an identifier, a list of components and an optional form in which the sequence of characters to which it corresponds can be copied.

The textual units relevant for Chinese NLP are the following three linguistically grounded levels: the word, chunk or phrase and sentences levels. Due to due to the specific writing of Chinese, where there is no space, the token level is irrelevant (see section 4.5).

4.2.1 Tokens

a. Definition

Tokens are the fundamental textual units in the Alvis text processing line. This segmentation is not linguistically grounded. It serves no other purpose but to provide a starting point from which to implement further segmentation.

This level of annotation follows the recommendations of the TC37SC4/TEI workgroup. However, this workgroup proposes to insert pointer mark-up (“TEI element ptr) in the textual signal to mark the token boundaries whereas we refer to the character offset.

To simplify further processing, we distinguish different types of tokens:

- Alphabetical tokens: sequences of letters (a-z and A-Z) including accented characters;
- Numerical tokens: sequences of digits (0-9);
- Separating tokens: sequence of separator characters (space, return…);
- Symbolic tokens: any other character.

The tokenisation is the first stage of text analysis. Tokens are numbered from 1 for the first token. All others annotations refer directly or indirectly to that token numbering.

b. Simplified examples

For readability reasons, the following examples are given with traditional inserted annotation instead of stand-off annotations. The slash represents the tokens boundaries (note that blanks are tokens). The actual Alvis DTD for the encoding of the token level is given below.

English example

```
/Transcription/ /of/ /the/ /cotB/,/ /cotC/,/ /and/ /cotX/ /genes/ /by/ /final/
/sigma/(/K/)/ /RNA/ /polymerase/ /is/ /activated/ /by/ /a/ /small/,/ /DNA/-/binding/
/protein/ /called/ /GerE/./
```

French example

```
/Toutes/ /ces/ /méthodes/ /permettaient/ /une/ /opacification/ /des/ /artères/
/coronariennes/./
```

c. Alvis format for token annotations

Alvis DTD for the token level

```
<!-- ============================================================ -->
<!--                    TOKEN LEVEL                               -->
<!-- ============================================================ -->
<!ELEMENT token_level (token_sep | token_alpha | token_symb | token_num )+          >

<!--                    separator token                           -->
<!ELEMENT  token_sep    (id, content, from, to)                     >

<!--                    alphanumeric token                        -->
<!ELEMENT  token_alpha  (id, content, from, to)                     >

<!--                    symbol token                              -->
<!ELEMENT  token_symb   (id, content, from, to)                     >

<!--                    numeric token                             -->
<!ELEMENT  token_num    (id, content, from, to)                     >

<!--                    content of the token                      -->
<!ELEMENT  content      (#PCDATA)                                   >

<!--                    start offset of the token                 -->
<!ELEMENT  from         (#PCDATA)                                   >

<!--                    end offset of the token                   -->
<!ELEMENT  to           (#PCDATA)                                   >
```

Example of Alvis annotation for the tokens

```
    <c_alpha>
      <cont>Combined</cont>
      <from>37</from>
```

```
      <id>token2</id>
      <to>44</to>
    </c_alpha>
    <c_sep>
      <cont> </cont>
      <from>45</from>
      <id>token3</id>
      <to>45</to>
    </c_sep>
    <c_alpha>
      <cont>action</cont>
      <from>46</from>
      <id>token4</id>
      <to>51</to>
    </c_alpha>
```

4.2.2 Words

a. Definition

Words are the basic linguistic units. They are made of tokens: every word is made of one or several tokens, numeric, alphabetic or symbolic. Words may contain spaces (i.e. "B. Subtilis" in biology or “pomme de terre” in French).

However, certain character strings are not trivially split into words, for example "doesn't" is made of the words "does" and "not", which do not appear as such. In such a case, two words (*does* and *not*) are created independently from the corresponding tokens (*doesn*, the apostrophe (*'*) and *t*).

b. Simplified examples

In the following examples, words are delimited by square brackets. Note that neither the punctuation marks nor the blanks are words. This segmentation can be compared to the tokenisation presented in the section above. The real Alvis format for the encoding of words is presented below.

English example

```
[Transcription] [of] [the] [cotB], [cotC], [and] [cotX] [genes] [by] [final] [sigma(K)]
[RNA] [polymerase] [is] [activated] [by] [a] [small], [DNA-binding]  [protein] [called]
[GerE].
```

French example

```
[Toutes] [ces] [méthodes] [permettaient] [une] [opacification] [de] [les]
[artères] [coronariennes].
```

c. Alvis format for word annotations

Alvis DTD for the word level

```
<!-- ============================================================ -->
<!--                    WORD LEVEL                                -->
<!-- ============================================================ -->
<!ELEMENT  word_level   (word)+                                     >

<!--                    word                                      -->
<!ELEMENT  word         (id, list_refid_token, form*)               >

<!--                    id of the element                         -->
<!ELEMENT  id           (#PCDATA)                                   >

<!--                    list of the tokens which compose the words -->
<!ELEMENT  list_refid_token
```

```
                        (refid_token)+                                  >

<!--                    token id, part of the words                 -->
<!ELEMENT  refid_token  (#PCDATA)                                       >

<!--                          form                                      -->
<!ELEMENT  form    (#PCDATA)                                            >
```

Example of Alvis annotation for the words

```
<word>
  <id>word43</id>
  <list_refid_token>
    <refid_token>token78</refid_token>
     </list_refid_token>
     <form>polymerase</form>
</word>
<word>
  <id>word44</id>
  <list_refid_token>
    <refid_token>token80</refid_token>
     </list_refid_token>
     <form>is</form>
</word>
<word>
  <id>word45</id>
  <list_refid_token>
    <refid_token>token82</refid_token>
     </list_refid_token>
     <form>activated</form>
</word>
```

4.2.3 Phrases

a. Definition

A phrase is a group of words (or a single word) that functions as a syntactic unit. It is composed of a head (the main part of the phrase) and of optional modifier(s) that can be words or phrases. The syntactic properties of a phrase are derived from its head (a Noun for a Noun Phrase, an Adjective for an Adjectival Phrase, etc.).

At the phrasal level described here, we only delimit the unit and no syntactic category is assigned to the phrase (the properties of the syntactic units are described in section 4.3).

b. Simplified examples

To illustrate the definition above, we give some simplified examples in which phrases are delimited with inserted brackets. The Alvis format for phrase delimitation is described below.

English

```
[During [sporulation of Bacillus subtilis]], [[spore coat proteins] [encoded [by [cot
genes]]]] [are expressed [in [the mother cell]] and [deposited [on [the forespore]]]].
[Transcription [of [the cotB, cotC, and cotX genes]] [by [final [sigma(K) RNA
polymerase]]] [is activated [by [a small, [DNA-binding protein] [called GerE]]]].
[The promoter region [of [each [of [these genes]]]]] [has [two [GerE binding sites]]].
```

French

```
[Toutes ces méthodes] [permettaient [une opacification des [artères coronariennes]]].
```

c. Alvis format for phrase annotations

Alvis DTD for the phrase level

```
<!-- ============================================================ -->
<!--                    PHRASE LEVEL                              -->
<!-- ============================================================ -->
<!ELEMENT  phrase_level
                        (phrase)+                                  >

<!--                    phrase                                    -->
<!ELEMENT  phrase      (id, list_refid_components, form*)          >

<!--                    list_refid_components                     -->
<!ELEMENT  list_refid_components
                        (refid_word | refid_phrase)+               >

<!--                    refid_phrase                             -->
<!ELEMENT  refid_phrase
                        (#PCDATA)                                  >
```

Example of Alvis annotation for the words

```
<c_phrase>
       <id>phrase1</id>
       <list_ref_id_components>
              <refid_word_or_phrase>word1</refid_word_or_phrase>
              <refid_word_or_phrase>word2</refid_word_or_phrase>
              <refid_word_or_phrase>phrase2</refid_word_or_phrase>
       </list_refid_components>
```

4.2.4 Semantic units

a. Definition

The semantic units are the textual units that are considered as significant on the semantic point of view. They can be either:

- Named entities that refer to well identified domain entities (often designated by proper names but not always)
- Terms that are the expressions referring to the concepts specific to the domain of the text.
- Undefined: other types of relevant semantic units can be identified, even if their semantic status is not stated.

b. Simplified examples

We give below some examples of terms issued from the MeSH[2] whose terminology can be used to tag documents. This is an example of the terminological resource that WP6 may provide to WP5 and that can be exploited to tag terms in documents.

```
abl gene
gene expression regulation
adrenergic uptake inhibitors
enzyme inhibitor, enzyme inhibitors
transcription
transcription factor, transcription factors
adenosine
```

In the examples below, the names entities and terms are tagged as XML-like inserted mark-up. For sake of readability, named entities are written in bold faces and terms are underlined. The real Alvis format for the encoding of semantic units is given in the next subsection.

[2] http://www.nlm.nih.gov/mesh/meshhome.html

Simplified example (English – terms and named entities in biological domain)

```
During sporulation of <named entity>Bacillus subtilis</named entity>, <term>spore coat
proteins</term> encoded by <term><named entity>cot</named entity> genes</term> are
expressed in the <term>mother cell</term> and deposited on the <term>forespore</term>.
Transcription of the <named entity>cotB</named entity>, <named entity>cotC</named entity
>, and <named entity>cotX</named entity> genes by final <named entity>sigma(K)</named
entity> <term>RNA polymerase</term> is activated by a small, <term>DNA-binding
protein</term> called <named entity>GerE</named entity>. The <term>promoter region</term>
of each of these genes has two <named entity>GerE</named entity> <term>binding
sites</term>.
```

Simplified example (English – named entities in general domain)

```
With heavy rains flooding northern <named entity>France</named entity>, a cartoon in
L'Express last week showed <named entity>Mr. Chirac</named entity>, the Gaullist leader,
trapped on a roof waiting to be rescued as <named entity>Prime Minister</named entity>
<named entity>Edouard Balladur</named entity> nonchalantly walked away across the water.
```

Simplified example (French – named entities)

```
12289. <named entity>Rhodia</named entity> cède des activités d'<named entity>Albright &
Wilson</named entity>. Le groupe cède les <named entity>surfactants Europe</named entity>
d'<named entity>Albright & Wilson</company> à <company>Huntsman International</named
entity>.
SOC <named entity>2001-03-01<named entity> <named entity>09:42:00<named entity>.000.
Dans le cadre du recentrage de ses activités, <named entity>Rhodia</named entity> a
annoncé ce matin la signature d'un accord définitif de cession des <named
entity>activités européennes surfactants</named entity> d'<named entity>Albright &
Wilson</named entity>.
```

Simplified example (French – terms in medical domain)

```
Toutes ces méthodes permettaient une <term>opacification des artères
coronariennes</term>.
```

c. Alvis format for semantic_unit annotations

Alvis DTD for the semantic_unit level

```
<!-- ============================================================ -->
<!--                    SEMANTIC UNIT                             -->
<!-- ============================================================ -->
<!--                    Named entities and terms                  -->
<!ELEMENT  semantic_unit
                       (named_entity | term | undefined)+                      >

<!--                    named entity                              -->
<!ELEMENT  named_entity (id, refid_phrase, form*,
                         canonical_form*, named_entity_type)          >

<!ELEMENT  list_refid_semantic_unit (refid_semantic_unit)+           >
<!ELEMENT  refid_semantic_unit (#PCDATA)                             >

<!--                    named_entity_type                         -->
<!ELEMENT  named_entity_type
                       (#PCDATA)                                     >

<!--                    list_refid_phrase                        -->
<!ELEMENT  list_refid_phrase
                       (refid_word | refid_phrase)+                  >
```

```
<!--                    term                                        -->
<!ELEMENT  term         (id, refid_phrase | refid_word, form*,
                         canonical_form*)                            >
```

Example of Alvis annotation for the semantic units

```
<semantic_unit>
       <term>
              <id>sem_unit1</id>
              <refid_phrase>phrase10</refid_phrase>
              <form>transcription factors</form>
              <canonical_form>transcription factor</canonical_form>
       </term>
       <term>
              <id>term2</id>
              <refid_word>word15</refid_word>
              <form>adenosine</form>
              <canonical_form>adenosine</canonical_form>
       </term>
</semantic_unit>
```

4.2.5 Sentences

a. Definition

The sentences correspond to a traditional textual unit. They usually start from a word with a capital initial character and ends with a period. However various other types of sentences can be encountered in texts. In the Alvis linguistic annotation format, we consider that titles, some list items and captions are sentences.

b. Simplified examples

In the following, the sentences are identified by inserted brackets for sake of simplicity. The real Alvis format for sentence delimitation is given below.

```
[During sporulation of Bacillus subtilis, spore coat proteins encoded by cot genes are
expressed in the mother cell and deposited on the forespore.]
[Transcription of the cotB, cotC, and cotX genes by final sigma(K) RNA polymerase is
activated by a small, DNA-binding protein called GerE.]
[The promoter region of each of these genes has two GerE binding sites.]
```

c. Alvis format for sentence annotation

Alvis DTD for the sentence level

```
<!-- ============================================================ -->
<!--                    SENTENCE LEVEL                            -->
<!-- ============================================================ -->
<!ELEMENT  sentence_level (sentence)+                               >

<!--                    sentence                                    -->
<!ELEMENT  sentence     (id, refid_start_token, refid_end_token,
                         form*)                                     >

<!--                    Reference of the token at the beginning     -->
<!--                    of the sentence                             -->
<!ELEMENT  refid_start_token
                        (#PCDATA)                                   >

<!--                    Reference of the token at the end           -->
<!--                    of the sentence                             -->
<!ELEMENT  refid_end_token
```

```
                        (#PCDATA)                                   >

<!--                    list of the words which compose the word    -->
<!ELEMENT  list_refid_word
                        (refid_word)+                               >

<!--                    word id, part of the word                   -->
<!ELEMENT  refid_word  (#PCDATA)                                    >
```

Example of Alvis annotation for the sentences

```
<sentence_level>
        <sentence>
                <id>sentence1</id>
                <refid_start_token>token1</refid_start_token>
                <refid_end_token>token26</refid_end_token>
                <list_refid_word>
                        <refid_word>word1</refid_word>
                        <refid_word>word2</refid_word>
                        <refid_word>word3</refid_word>
                        <refid_word>word4</refid_word>
                        <refid_word>word5</refid_word>
                        <refid_word>word6</refid_word>
                        <refid_word>word7</refid_word>
                        <refid_word>word8</refid_word>
                        <refid_word>word9</refid_word>
                        <refid_word>word10</refid_word>
                        <refid_word>word11</refid_word>
                        <refid_word>word12</refid_word>
                        <refid_word>word13</refid_word>
                </list_refid_word>
                <form>The promoter region of each of these genes has two GerE binding
sites.</form>
        </sentence>
</sentence_level>
```

4.3 Morpho-syntactic properties

Various morpho-syntactic properties can be associated with the words or the phrases: the lemmas, the stems, the morpho-syntactic features and syntactic relations.

4.3.1 Lemmas

a. Definition

Lemma is the canonical form of a word. See section 3.2 for a more detailed presentation

b. Simplified examples

We give below examples of lemma of English words.

word	lemma
sporulation	sporulation
proteins	protein
are	be
expressed	express

For illustration purpose, in the following sentence, each word is associated with its lemma in XML-like inserted mark-up and the lemmas are written in bold font. The real Alvis format for lemma annotation is described below.

```
During<lemma>during</lemma> sporulation<lemma>sporulation</lemma> of<lemma>of</lemma>
Bacillus subtilis<lemma>Bacillus subtilis</lemma>, spore<lemma>spore</lemma>
coat<lemma>coat</lemma> proteins<lemma>protein</lemma> encoded<lemma>encode</lemma>
by<lemma>by</lemma> cot<lemma>cot</lemma> genes<lemma>gene</lemma> are<lemma>be</lemma>
expressed<lemma>express</lemma> in<lemma>in</lemma> the<lemma>the</lemma>
mother<lemma>mother</lemma> cell<lemma>cell</lemma> and<lemma>and</lemma>
deposited<lemma>deposit</lemma> on<lemma>on</lemma> the<lemma>the</lemma>
forespore<lemma>forespore</lemma> .
```

c. Alvis format for lemma annotations

Alvis DTD for the lemma level

```
<!-- ============================================================ -->
<!--                    LEMMA LEVEL                               -->
<!-- ============================================================ -->
<!ELEMENT  lemma_level  (lemma)+                                   >

<!--                    lemma                                     -->
<!ELEMENT  lemma        (id, canonical_form+, refid_word,
                         form*)                                    >

<!--                    canonical form of the word                -->
<!--                    corresponding to the lemma                -->
<!ELEMENT  canonical_form
                        (#PCDATA)                                  >
```

Example of Alvis annotation for the lemmas

```
<lemma_level>
        <lemma>
                <id>lemma7</id>
                <canonical_form>this</canonical_form>
                <refid_word>word7</refid_word>
                <form>these</form>
        </lemma>
        <lemma>
                <id>lemma8</id>
                <canonical_form>gene</canonical_form>
                <refid_word>word8</refid_word>
                <form>genes</form>
        </lemma>
        <lemma>
                <id>lemma9</id>
                <canonical_form>have</canonical_form>
                <refid_word>word9</refid_word>
                <form>has</form>
        </lemma>
        <lemma>
                <id>lemma10</id>
                <canonical_form>two</canonical_form>
                <refid_word>word10</refid_word>
                <form>two</form>
        </lemma>
        <lemma>
                <id>lemma11</id>
                <canonical_form>GerE</canonical_form>
                <refid_word>word11</refid_word>
                <form>GerE</form>
        </lemma>
        <lemma>
                <id>lemma12</id>
                <canonical_form>bind</canonical_form>
```

```
                <refid_word>word12</refid_word>
                <form>binding</form>
        </lemma>
        <lemma>
                <id>lemma13</id>
                <canonical_form>site</canonical_form>
                <refid_word>word13</refid_word>
                <form>sites</form>
        </lemma>
</lemma_level>
```

4.3.2 Stems

a. Definition

A stem is considered here as the character string corresponding to a word without all or part of its suffixes, eventually with one or few additional characters. The stem is not defined as a linguistic unit but as an arbitrary character string. A given word is supposed to have a single stem, even if several candidates could be considered.

b. Simplified examples

The table below give examples of stems for some English words.

word	stem
eventually	eventual
eventual	eventual
expression	express
expressed	express

For illustration purpose, in the following sentence, each word is associated with its stem in XML-like inserted mark-up and the stems are written in bold font. The real Alvis format for stem annotation is described below.

```
During<stem>dur</stem> sporulation<stem>sporulat</stem> of<stem>of</stem> Bacillus
subtilis<stem>Bacillus subtilis</stem>, spore<stem>spore</stem> coat<stem>coat</stem>
proteins<stem>protein</stem> encoded<stem>encode</stem> by<stem>by</stem>
cot<stem>cot</stem> genes<stem>gene</stem> are<stem>be</stem>
expressed<stem>express</stem> in<stem>in</stem> the<stem>the</stem>
mother<stem>mother</stem> cell<stem>cell</stem> and<stem>and</stem>
deposited<stem>deposit</stem> on<stem>on</stem> the<stem>the</stem>
forespore<stem>forespore</stem> .
```

c. Alvis format for stem annotations

Alvis DTD for the stem level

```
<!-- ============================================================ -->
<!--                    STEM LEVEL                                -->
<!-- ============================================================ -->
<!ELEMENT  stem_level   (stem)+                                    >

<!--                    stem                                      -->
<!ELEMENT  stem         (id, stem_form+, refid_word,
                         form*)                                    >

<!--                    stem form                                 -->
<!ELEMENT  stem_form    (#PCDATA)                                  >
```

Example of Alvis annotation for the stems

```
<stem_level>
```

```
        <stem>
                <id>stem7</id>
                <stem_form>this</stem_form>
                <refid_word>word7</refid_word>
                <form>these</form>
        </stem>
        <stem>
                <id>stem8</id>
                <stem_form>gene</stem_form>
                <refid_word>word8</refid_word>
                <form>genes</form>
        </stem>
        <stem>
                <id>stem9</id>
                <stem_form>ha</stem_form>
                <refid_word>word9</refid_word>
                <form>has</form>
        </stem>
        <stem>
                <id>stem10</id>
                <stem_form>two</stem_form>
                <refid_word>word10</refid_word>
                <form>two</form>
        </stem>
        <stem>
                <id>stem11</id>
                <stem_form>GerE</stem_form>
                <refid_word>word11</refid_word>
                <form>GerE</form>
        </stem>
        <stem>
                <id>stem12</id>
                <stem_form>bind</stem_form>
                <refid_word>word12</refid_word>
                <form>binding</form>
        </stem>
        <stem>
                <id>stem13</id>
                <stem_form>site</stem_form>
                <refid_word>word13</refid_word>
                <form>sites</form>
        </stem>
</stem_level>
```

4.3.3 Morpho-syntactic features

a. Definition

A syntactic unit may have a syntactic category and morphologic features:

- A syntactic category is either a phrasal category, such as noun phrase or verb phrase, which can be decomposed into smaller syntactic categories, or a lexical category (also called "part of speech" or POS category) such as noun or verb, which cannot be further decomposed[3]
    - Syntactic categories (i.e. lexical categories or parts of speech) are assigned to words by the morpho-syntactic tagger. Depending on the tagger used, the number and granularity of syntactic categories vary. The tags always express syntactic category (Noun, Verb, Adverb, Adjective, Conjunction, etc.) and sometimes give information about morphological features, such as Gender and Number for nouns, or Tense and Voice for verbs. The total number of tags in a tag set ranges from about 30 to 350, depending on the level of information desired and the language of the corpus. The tag set used at

[3] http://en.wikipedia.org/wiki/Syntactic_categories

present is the Penn TreeBank and contains several tens of tags for English, as the same for French. The potential morphological features that can be associated with a given syntactic unit depend on its syntactic category, as shown in the tables below.

The tags can be ambiguous or not. Tags are generally assigned to the words of a sentence according to their context. Context is needed because a significant number of words can receive two or more syntactic tags when isolated from their context (e.g. "show" is either a noun or a verb without context, but it is definitely a noun when preceded by "a" and a verb when followed by "me"). Some ambiguities still remain even in context, and, in this case, the tagger must have default behaviour.

- Taggers generally use a dictionary and exploit morphological rules to tag unknown words. Results on general language have now reached a very good level for English, Slovene and French. Tagging a domain specific language is trickier and needs the training of the tagger when it is possible.

- Syntactic parsers also give syntactic categories information at a phrasal level. It identifies the head of each of the phrases that have been delimited by the parser and propagates its category features to the phrasal level. For instance, a phrase with a head carrying a Noun category will be assigned the Noun Phrase category.

b. Simplified examples

To illustrate the definitions above, we give some examples of sentences tagged according to the nomenclature given in the tables below. These tags combine a syntactic category and some morphological features. As mentioned above, the exact nomenclature depends on the NLP tools that are used for tagging and parsing.

Word syntactic categories

Syntactical features		Morphological features		Grammatical features		List of complete tags used in the examples
Tag	***Description***	Tag	***Description***	Tag	***Description***	
ADJ	Adjective	m / f s / p	masculine / feminine, singular / plural			ADJ, ADJfp
CC	Coordination					CC
DT	Determiner	m / f s / p	masculine / feminine, singular / plural			DT, DTs, DTp, DTfs
N	Noun	m / f s / p	masculine / feminine, singular / plural			Ns, Np, N, Nfp, Nfs
NUM	Number	s / p	singular / plural			NUMp
PN	Proper Noun					PN
PREP	Preposition					PREP
PREPDT	Preposition + Determiner	m / f s / p	masculine / feminine, singular / plural			PREPDTp
PUNCT	Punctuation					PUNCT
V	Verb			aux / ed pst / imp	auxiliary / ed-form present / preterit	Vaux, Ved, Vpst, Vimp

Phrase syntactic categories

Tag	Description
NP	Noun Phrase
VP	Verb Phrase
PP	Preposition Phrase

The following examples are simplified ones. The tags are given as subscripts associated with the preceding word or the bracket delimited phrase. The actual Alvis format for encoding morpho-syntactic features is given below.

English

$[\text{During}_{\text{PREP}}\ [\text{sporulation}_{\text{Ns}}\ \text{of}_{\text{PREP}}\ \text{Bacillus subtilis}_{\text{PN}}]_{\text{NP}}]_{\text{PP}},\ [[\text{spore}_{\text{Ns}}\ \text{coat}_{\text{Ns}}\ \text{proteins}_{\text{Np}}]_{\text{NP}}\ [\text{encoded}_{\text{Ved}}\ [\text{by}_{\text{PREP}}\ [\text{cot genes}]_{\text{PN}}]_{\text{PP}}]_{\text{VP}}]_{\text{NP}}\ [\text{are}_{\text{Vaux}}\ \text{expressed}_{\text{Ved}}\ [\text{in}_{\text{PREP}}\ [\text{the}_{\text{DTs}}\ \text{mother}_{\text{Ns}}\ \text{cell}_{\text{Ns}}]_{\text{NP}}]_{\text{PP}}\ \text{and}_{\text{CC}}\ [\text{deposited}_{\text{Ved}}\ [\text{on}_{\text{PREP}}\ [\text{the}_{\text{DT}}\ \text{forespore}_{\text{Ns}}]_{\text{NP}}]_{\text{PP}}]_{\text{VP}}].$

French

$[\text{Toutes}_{\text{ADJfp}}\ \text{ces}_{\text{DTfp}}\ \text{méthodes}_{\text{Nfp}}]_{\text{NPfp}}\ [\text{permettaient}_{\text{Vimp}}\ [\text{une}_{\text{DTfs}}\ \text{opacification}_{\text{Nfs}}\ [\text{des}_{\text{DTfp}}\ \text{artères}_{\text{Nfp}}\ \text{coronariennes}_{\text{ADJfp}}]_{\text{NPfp}}]_{\text{NPfs}}]_{\text{VP}}.$

c. Alvis format for syntactic category annotations

Syntactic categories of words, as well as phrases, are annotated at the morphosyntactic_features level.

Alvis DTD for the morphosyntactic_features level

```
<!-- ============================================================ -->
<!--                    MORPHOSYNTACTIC FEATURES LEVEL              -->
<!-- ============================================================ -->

<!ELEMENT  morphosyntactic_features_level
                       (morphosyntactic_features)+                  >

<!--                   morphosyntactic_features                     -->
<!ELEMENT  morphosyntactic_features
                       (id, refid_word|refid_phrase, tense*, aspect*,
                        person*, number*, voice*, gender*,
                        syntactic_category, form*)            >
<!--                   tense                                        -->
<!ELEMENT  tense       (#PCDATA)                                     >
<!--                   aspect                                       -->
<!ELEMENT  aspect      (#PCDATA)                                     >
<!--                   number                                       -->
<!ELEMENT  number      (#PCDATA)                                     >
<!--                   person                                       -->
<!ELEMENT  person      (#PCDATA)                                     >
<!--                   voice                                        -->
<!ELEMENT  voice       (#PCDATA)                                     >
<!--                   gender                                       -->
<!ELEMENT  gender      (#PCDATA)                                     >
<!--                   syntactic_category                           -->
<!ELEMENT  syntactic_category
                       (#PCDATA)                                     >
```

Example of Alvis annotation for the morphosyntactic features

```
expressed:
<morphosyntactic_features_level>
       <morphosyntactic_features >
              <id>msf1</id>
              <refid_word>word10</refid_word>
              <syntactic_category>VERB</syntactic_category>
                     <tense>past</tense>
              <aspect>simple</aspect>
              <person>third</person>
              <number>singular</number>
              <voice>active</voice>
              <form>expressed</form>
       </morphosyntactic_features>
       <morphosyntactic_features >
              <id>msf2</id>
              <refid_phrase>phrase24</refid_phrase>
              <syntactic_category>NOUN PHRASE</syntactic_category>
                     <number>plural</number>
              <form>spore proteins</form>
       </morphosyntactic_features>
</morphosyntactic_features_level>
```

4.3.4 Syntactic relations

a. Definition

A syntactic relation defines the role (or function) played by two words between one another. This relation is represented as a triplet: a relation type T, its head H (or governor) and its expansion E (or dependent, also called modifier or argument).

b. Simplified examples

To illustrate the definition above, we give two examples, one in English and one in French. The following table list 14 available relations. For each relation, we indicate its name, its description, the possible types of the head and the expansion elements it may connect to.

Name	Description	Head	Expansion
SUBJECT	subject - verb	Verb	Noun, Pronoun
INVSUBJ	subject – verb in reversed order sentences	Verb	Noun, Pronoun
SUJPAS	subject of a passive verb	Verb	Noun, Pronoun
AGTPAS	agent of a passive verb	Verb	Noun, Pronoun
OBJD	verb - object	Verb	Noun, Pronoun
VCOMP(X)	verb – complement introduced by a preposition X	Verb	Noun, Pronoun
ADV	adverb – verb	Verb	Adverb
PADV	adverb postposed to a verb	Verb	Adverb
NN	noun – noun	Noun1	Noun2
ADJ	adjective – noun	Noun	Adjective
PRADJ	noun - predicate adjective	Noun	Adjective
NCOMP(X)	noun - complement introduced by a preposition X	Noun	Noun, Pronoun, Verb
ADVADJ	adverb – adjective	Adjective	Adverb
ADVADV	adverb - adverb	Adverb1	Adverb2

In the examples below, each sentence comes with the list of its syntactic relations. The syntactic relation triplet is represented by (**type (head , expansion)**)

English

```
During sporulation of Bacillus subtilis, spore coat proteins encoded by cot genes are
expressed in the mother cell and deposited on the forespore.
NCOMPduring(express, sporulation)
NCOMPof (sporulation,bacillus)
ADJ (bacillus,subtilis)
NN (coat,spore)
NN (spore,protein)
SUJPAS (encode,protein)
SUJPAS (express,protein)
SUJPAS (deposit,protein)
VCOMPby (encode,gene)
AGTPAS (encode,gene)
NN (gene,cot)
VCOMPin (express,cell)
NN (cell,mother)
VCOMPon (deposit,forespore)
```

French

```
Toutes ces méthodes permettaient une opacification des artères coronariennes.
SUBJECT(permettre,méthode)
OBJECT(permettre,opacification)
NCOMPde(opacification,artère)
ADJ(artère,coronarienne)
```

c. Alvis format for syntactic relations annotations

Alvis DTD for the syntactic relation level

```
<!-- ============================================================ -->
<!--                    SYNTACTIC_RELATION_LEVEL                  -->
<!-- ============================================================ -->
<!ELEMENT  syntactic_relation_level
                        (syntactic_relation)+                      >

<!--                    syntactic_relation                        -->
<!ELEMENT  syntactic_relation
                        (id, syntactic_relation_type,
                         refid_head,
                         refid_modifier)                           >

<!--                    refid_head phrase or word                 -->
<!ELEMENT  refid_head
                        (refid_word | refid_phrase)                >

<!--                    refid_modifier phrase or word             -->
<!ELEMENT  refid_modifier
                        (refid_word | refid_phrase)                >


<!--                    syntactic_relation_type                   -->
<!ELEMENT  syntactic_relation_type
                        (#PCDATA)                                  >
```

Example of Alvis annotation for the syntactic relations

```
<syntactic_relation>
      <id>syntrel1</id>
      <syntactic_relation_type>NCOMPby</syntactic_relation_type>
      <refid_modified_phrase>word26</refid_modified_phrase>
      <refid_modifier_phrase>word35</refid_modifier_phrase>
    </syntactic_relation>
```

4.4 Semantic properties

Two types of semantic properties hold for semantic units: the semantic types and the semantic relations.

4.4.1 Semantic types

a. Definition

Semantic categories are attached to semantic units that can be words (registered as undefined units by the tagger), named entities or terms, as defined by conceptual hierarchies of the domain.

As mentioned in section 3.4, semantic typing may rely or not on syntactic parsing. In any case, the format of the document tagging will be the same. The tagging processes only differ.

b. Simplified examples

As an illustration, we attach semantic categories to relevant instances of biological concepts in the following sentence:

```
Transcription of the cotB, cotC, and cotX genes by final sigma(K) RNA polymerase is
activated by a small, DNA-binding protein called GerE.
```

Relevant concepts and their instances are given in the hierarchy as followed:

```
cotB -> gene -> biological_object
cotC -> gene -> biological_object
```

```
cotX -> gene -> biological_object
sigmaK -> transcription factor -> protein -> biological_object
GerE -> protein -> biological_object
```

For readability reasons, the annotations are given as inserted XML-like mark-up. The real Alvis format for the encoding of semantic types is given in the following subsection

The semantic tagging of example sentence will produce:

```
Transcription of the <category name="/biological_object/gene">cotB</category>, <category
name="/biological_object/gene">cotC</category>, and <category
name="/biological_object/gene"> cotX</category> genes by final <category
name="/biological_object/protein/transcription_factor">sigma(K)</category> RNA polymerase
is activated by a small, DNA-binding protein called <category
name="/biological_object/protein">GerE</category>.
```

c. Alvis format for the annotation of semantic categories

Semantic categories are attached to the semantic units and are introduced in the annotated document in the *semantic_feature_level* XML block. We have to provide the hierarchical path related to the instance and, in case of ambiguity, all alternative paths. The format of the hierarchical path is defined by WP2 specifications: a minimal sequence of ontology nodes, from specific to general, identifying without ambiguity the hierarchical path. A node is identified by a unique ID, according to the ontology. As an example, consider we have to tag "cat" in the phrase "cat disease" according to the following hierarchy (*IS-A* relations):

```
cat -> gene -> DNA -> biological object
cat -> mammalian -> specie
```

In that case, we want to tag "cat" as a mammalian. If "cat" is represented as a unique node in the ontology, the tagging of the phrase example is:

```
<category node"cat",node="mammalian">cat</category> disease
```

In ambiguous cases, when the right hierarchical path cannot be identified, we provide the various alternatives represented as multiple tags (a sequence of uniquely identified hierarchical paths, from most likely to unlikely).

Alvis DTD for the syntactic relation level

```
<!-- ============================================================ -->
<!--                    SEMANTIC_FEATURE_LEVEL                    -->
<!-- ============================================================ -->
<!ELEMENT  semantic_feature_level
                        (semantic_feature)+                        >

<!--                    semantic_feature                          -->
<!ELEMENT  semantic_feature
                        (id, semantic_category, refid_semantic_unit) >

<!--                    semantic_category                         -->
<!ELEMENT  semantic_category
                        (list_refid_ontology_node)+                >

<!--                    list_refid_ontology_node                  -->
<!ELEMENT  list_refid_ontology_node
                        (refid_ontology_node)+                     >

<!--                    refid_ontology_node                       -->
<!ELEMENT  refid_ontology_node
                        (#PCDATA)                                  >
```

Example in the Alvis format

The following example (in stand-off annotation) corresponds to the same sentence as above: `Transcription of the cotB, cotC, and cotX genes by final sigma(K) RNA polymerase is activated by a small, DNA-binding protein called GerE.`

```
<semantic_feature_level>

    <!-- term tagging -->
    <semantic_feature>
      <id>sf5</id>
      <semantic_category>
        <list_refid_ontology_node>
          <refid_ontology_node>node25</refid_ontology_node>
        </list_refid_ontology_node>
      </semantic_category>
      <refid_semantic_unit>term7</refid_semantic_unit>
    </semantic_feature>
    <semantic_feature>
      <id>sf6</id>
      <semantic_category>
        <list_refid_ontology_node>
          <refid_ontology_node>node6</refid_ontology_node>
        </list_refid_ontology_node>
      </semantic_category>
      <refid_semantic_unit>term8</refid_semantic_unit>
    </semantic_feature>

    <!-- named entity tagging -->
    <semantic_feature>
      <id>sf10</id>
      <semantic_category>
        <list_refid_ontology_node>
          <refid_ontology_node>node7</refid_ontology_node>
        </list_refid_ontology_node>
      </semantic_category>
      <refid_semantic_unit>named_entity3</refid_semantic_unit>
    </semantic_feature>
    <semantic_feature>
      <id>sf11</id>
      <semantic_category>
        <list_refid_ontology_node>
          <refid_ontology_node>node8</refid_ontology_node>
        </list_refid_ontology_node>
      </semantic_category>
      <refid_semantic_unit>named_entity4</refid_semantic_unit>
    </semantic_feature>
    <semantic_feature>
      <id>sf12</id>
      <semantic_category>
        <list_refid_ontology_node>
          <refid_ontology_node>node9</refid_ontology_node>
        </list_refid_ontology_node>
      </semantic_category>
      <refid_semantic_unit>named_entity5</refid_semantic_unit>
    </semantic_feature>
    <semantic_feature>
      <id>sf13</id>
      <semantic_category>
        <list_refid_ontology_node>
          <refid_ontology_node>node10</refid_ontology_node>
        </list_refid_ontology_node>
      </semantic_category>
      <refid_semantic_unit>named_entity6</refid_semantic_unit>
    </semantic_feature>
    <semantic_feature>
      <id>sf14</id>
      <semantic_category>
        <list_refid_ontology_node>
          <refid_ontology_node>node12</refid_ontology_node>
```

```
        </list_refid_ontology_node>
      </semantic_category>
      <refid_semantic_unit>named_entity7</refid_semantic_unit>
    </semantic_feature>

    <!-- undefined unit tagging -->
    <!-- Transcription -->
    <semantic_feature>
      <id>sf17</id>
      <semantic_category>
        <list_refid_ontology_node>
          <refid_ontology_node>node26</refid_ontology_node>
        </list_refid_ontology_node>
      </semantic_category>
      <refid_semantic_unit>su1</refid_semantic_unit>
    </semantic_feature>
</semantic_feature_level>
```

4.4.2 Semantic relations

a. Definition

Two types of semantic relations can be encoded in documents: anaphoric relations and domain specific relations. They are attached to semantic units (either named entities, terms or undefined semantic units) which may corresponds to words or phrases.

Anaphoric relations

An anaphoric relation establishes a coreference link between two semantic units corresponding to two textual units, the anaphoric expression (or anaphor) and another textual unit occurring either prior or posterior to the anaphoric expression.

Domain specific relations

The domain specific relations identify relations occurring between instances of the ontological concepts in the document. The instances are semantic units. Section 6.11 discusses two possible scenarios whereas the domain specific relations are directly provided by an ontology or are deduced from text through pattern matching of information extraction rules (see also section 3.4).

b. Simplified examples

In the examples below, for sake of readability, the annotations are given as inserted mark-up. The real Alvis format for semantic relations is given in the following subsection.

The examples below make explicit all domain specific relations pertaining to the genic interactions explicitly mentioned in the following sentence as well as the anaphoric relations in the following sentence:

```
Transcription of the cotB, cotC, and cotX genes by final sigma(K) RNA polymerase is
activated by a small, DNA-binding protein called GerE.
```

Domain specific relations (genic interactions)

```
<category name="genic_interaction/positive_genic_interaction>Transcription of the
<category name="/biological_object/gene">cotB</category>, <category
name="/biological_object/gene">cotC</category>, and <category
name="/biological_object/gene"> cotX</category> genes by final <category
name="/biological_object/protein/transcription_factor">sigma(K)</category> RNA polymerase
is activated by a small, DNA-binding protein called <category
name="/biological_object/protein">GerE</category>
<dsr name="domain_specific_relation/binary_relation/genic_interaction/agent><agent
name="sigmaK"><object name="Transcription">
```

```
<dsr name="domain_specific_relation/binary_relation/genic_interaction/object><agent
name="Transcription"><object name="cotB">."
<dsr name="domain_specific_relation/binary_relation/genic_interaction/object><agent
name="Transcription"><object name="cotC">."
<dsr name="domain_specific_relation/binary_relation/genic_interaction/object><agent
name="Transcription"><object name="cotX">.
```

Coreference relations

```
Transcription of the <category name="/anaphora relation/antecedent", relation
id=1>cotB</category name>, <category name="/anaphora relation/antecedent", relation
id=1>cotC</category name>, and <category name="/anaphora relation/antecedent", relation
id=1>cotX</category name> genes by final sigma(K) RNA polymerase is activated by a small,
DNA-binding protein called GerE. The promoter region of each of <category name="/anaphora
relation/pronoun", relation id=1>these</category name> genes has two GerE binding sites.
```

c. Alvis format for the annotation of semantic relations

The semantic relations are attached to semantic units. These relations have an identifier, a type (either a domain specific relation type or an anaphora relation type). They are introduced in the annotated document in the *semantic_relation_level* XML block. The format allows representing relations of arbitrary arity between semantic unit IDs. A relation is seen as a semantic relation tag, followed by a sequence of arguments; the arity of the relation and the type of its arguments are deduced from the sequence of arguments. For anaphora relations, the role of these arguments is made explicit (either anaphor or antecedent).

Alvis DTD for the semantic relation level

```
<!-- ============================================================ -->
<!--                    DOMAIN_SPECIFIC_RELATION_LEVEL            -->
<!-- ============================================================ -->
<!ELEMENT  domain_specific_relation_level
                        (domain_specific_relation)+                >

<!--                    domain_specific_relation                  -->
<!ELEMENT  domain_specific_relation
                        (id, domain_specific_relation_type,
                         list_refid_semantic_unit)   >

<!--                    domain_specific_relation_type             -->
<!ELEMENT  domain_specific_relation_type
                        (#PCDATA)                                    >


<!-- ============================================================ -->
<!--                    ANAPHORA_RELATION_LEVEL                    -->
<!-- ============================================================ -->
<!ELEMENT  anaphora_relation_level
                        (anaphora_relation)+                         >
<!--                    anaphora_relation                          -->
<!ELEMENT  anaphora_relation
                        (id, anaphora_relation_type,
                         anaphora, antecedent)                       >

<!ELEMENT  antecedent
                        (list_refid_semantic_unit)+                  >
<!ELEMENT  anaphora
                        (refid_semantic_unit)                        >


<!--                    anaphora_relation_type                     -->
<!ELEMENT  anaphora_relation_type
                        (#PCDATA)                                    >
```

Example in the Alvis format

In the following example (in Alvis stand-off annotation format) all the genic interactions between genes and proteins in the following sentence are made explicit:

```
transcription of the cotB, cotC, and cotX genes by final sigma(K) RNA polymerase is
activated by a small, DNA-binding protein called GerE.
```

```
<semantic_relation_level>
  <semantic_relation>
    <id>sr1</id>
    <semantic_relation_type>genic_agent</semantic_relation_type>
    <list_refid_semantic_unit>
      <refid_semantic_unit>named_entity6</refid_semantic_unit>
      <refid_semantic_unit>term7</refid_semantic_unit>
    </list_refid_semantic_unit>
  </semantic_relation>
  <semantic_relation>
    <id>sr2</id>
    <semantic_relation_type>genic_target</semantic_relation_type>
    <list_refid_semantic_unit>
      <refid_semantic_unit>named_entity3</refid_semantic_unit>
      <refid_semantic_unit>term7</refid_semantic_unit>
    </list_refid_semantic_unit>
  </semantic_relation>
  <semantic_relation>
    <id>sr3</id>
    <semantic_relation_type>genic_target</semantic_relation_type>
    <list_refid_semantic_unit>
      <refid_semantic_unit>named_entity4</refid_semantic_unit>
      <refid_semantic_unit>term7</refid_semantic_unit>
    </list_refid_semantic_unit>
  </semantic_relation>
  <semantic_relation>
    <id>sr4</id>
    <semantic_relation_type>genic_target</semantic_relation_type>
    <list_refid_semantic_unit>
      <refid_semantic_unit>named_entity5</refid_semantic_unit>
      <refid_semantic_unit>term7</refid_semantic_unit>
    </list_refid_semantic_unit>
  </semantic_relation>
  <semantic_relation>
    <id>sr5</id>
    <semantic_relation_type>genic_agent</semantic_relation_type>
    <list_refid_semantic_unit>
      <refid_semantic_unit>named_entity7</refid_semantic_unit>
      <refid_semantic_unit>su1</refid_semantic_unit>
    </list_refid_semantic_unit>
  </semantic_relation>
  <semantic_relation>
    <id>sr6</id>
    <semantic_relation_type>genic_target</semantic_relation_type>
    <list_refid_semantic_unit>
      <refid_semantic_unit>named_entity3</refid_semantic_unit>
      <refid_semantic_unit>su1</refid_semantic_unit>
    </list_refid_semantic_unit>
  </semantic_relation>
  <semantic_relation>
    <id>sr7</id>
    <semantic_relation_type>genic_target</semantic_relation_type>
    <list_refid_semantic_unit>
      <refid_semantic_unit>named_entity4</refid_semantic_unit>
      <refid_semantic_unit>su1</refid_semantic_unit>
    </list_refid_semantic_unit>
```

```
    </semantic_relation>
    <semantic_relation>
      <id>sr8</id>
      <semantic_relation_type>genic_target</semantic_relation_type>
      <list_refid_semantic_unit>
        <refid_semantic_unit>named_entity5</refid_semantic_unit>
        <refid_semantic_unit>su1</refid_semantic_unit>
      </list_refid_semantic_unit>
    </semantic_relation>
  </semantic_relation_level>
```

4.5 Textual annotations for Chinese

Due to the specific complexity of Chinese NLP, the annotations for Chinese differ from that of the other languages:

- The textual units relevant for Chinese NLP are the following three linguistically grounded levels: the word, chunk or phrase and sentence level.
- The categories associated with these units combine syntactic and semantic information.

As explained below (section 5.4), the Chinese NLP line is divided in three different modules. The following example shows the various levels of annotations produced for a given sentence.

Raw input sentence

2004 年美国总统大选，共和党总统候选人布什击败民主党总统候选人克里，竟选连任成功。 (Bush defeated Kerry and wined 2004 US elections)

Explanation: Most Chinese sentences are ended with period(.), question mark(?) or exclamation mark (!)

Output after stage 1: Word segmentation, POS tagging and basic name entity identification

2004 年/t 美国/ns 总统/n 大选/vn ，/w 共和党/n 总统/n 候选人/n 布什/nr 击败/v 民主/a 党/n 总统/n 候选人/n 克里/nr ，/w 竟选/v 连任/v 成功/a 。/w

Explanations: For sake of readability, in the example above, the annotation format is {<Word> / <POS tag>}. It should be easily transformed into Alvis linguistic annotation format in the future. Some brief descriptions of the POS tags used in the sentence are as follows:

Tag	description
t	noun of time
ns	proper location name
n	noun
vn	verb
nr	proper personal name
v	verb
a	adjective
w	punctuation

[tp 2004 年/t] [np 美国/ns 总统/n 大选/vn] ， /w [np-O 共和党/n] [np-H 总统/n 候选人/n 布什/nr] [vp 击败/v] [np-O 民主/a 党/n] [np-H 总统/n 候选人/n 克里 /nr] ， /w [vp 竞选/v 连任/v] [ap 成功/a] 。 /w

Output after stage 2 & 3: partial parsing + semantic type tagging

[tp 2004 年/t] [np 美国/ns 总统/n 大选/vn] ， /w [np-O 共和党/n] [np-H 总统/n 候选人/n 布什/nr] [vp 击败/v] [np-O 民主/a 党/n] [np-H 总统/n 候选人/n 克里 /nr] ， /w [vp 竞选/v 连任/v] [ap 成功/a] 。 /w

Explanations: The annotation format is { [<chunk tag>-<semantic type> {<Word> / <POS tag>}] }. Some brief descriptions of the chunk tags and semantic types are as follows:

Tag	description
tp	basic noun phrase for time
np	basic noun phrase
vp	basic verb phrase
ap	basic adjective phrase
O	organization
H	human

5 Architecture of the text processing line

This section describes the overall architecture of the WP5 NL processing line. For sake of simplicity, four different diagrams are presented for the different languages addressed in Alvis, but those diagrams reflect the similarity between the four NLP lines. The main differences concerns the

- The level of NLP analysis, which is deeper for English and to a less extend for French than for other languages (as explained in section 2.2 above): neither parsing or semantic tagging for Slovene, for instance;
- The fact that modularity is less fine-grained for Slovene and Chinese: some processing steps are grouped together for computational or linguistic reasons;

In the following figures, the diagrams represent the WP5 NL processing line for the various languages addressed in Alvis. The input comes from WP7 and the output is provided either to WP2 (production line) or to WP6 (acquisition line), as explained above (section 2.1).

The various NL processing modules composing the NLP line are represented as boxes. The description of these modules are given in section 6 below. Doted boxes refer to modules for which no definitive decision has been taken: whether they will be integrated in Alvis or not will depend on the first results on English (do they have an impact on IR?) and on the complexity of the rest of the NLP work.

The arrows represent the WP5 processing flow. The doted arrows represent alternative types of outputs that WP5 may produce. They correspond to the various types of annotated documents that WP5 may provide to WP6 and WP2.

5.1 English NL processing line

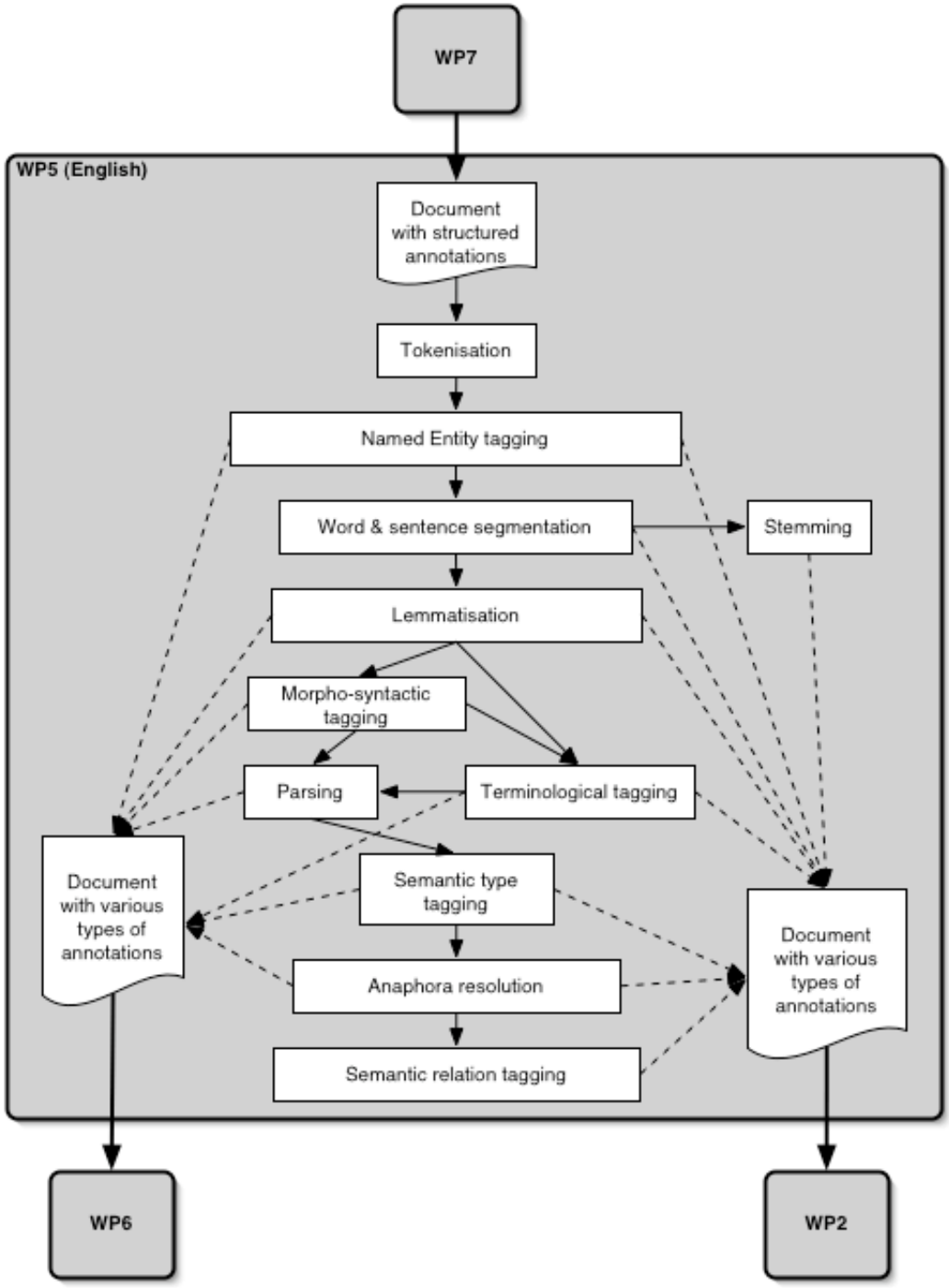


Figure 1 Architecture of the WP5 NL processing line for English.

This architecture is quite traditional but few points should be highlighted:

- The tokenisation is a first step to compute the first basic segmentation used for reference, as explained in section 4.2.1;
- As shown by the plain and dotted output arrows, many modules can be considered as a pre-processing step for further NLP or as a final step;

- The Named Entity tagging takes place very early in the NLP line because unrecognized named entities hinder most NLP steps, in many sublanguages;
- The stemming and lemmatisation are two alternative morphological normalisation steps, but stemming is only considered as a final step whereas lemmatisation is also used as a pre-processing step for syntactic and semantic processing;
- The terminological tagging can be considered as an aid for syntactic parsing;

5.2 French NL processing line

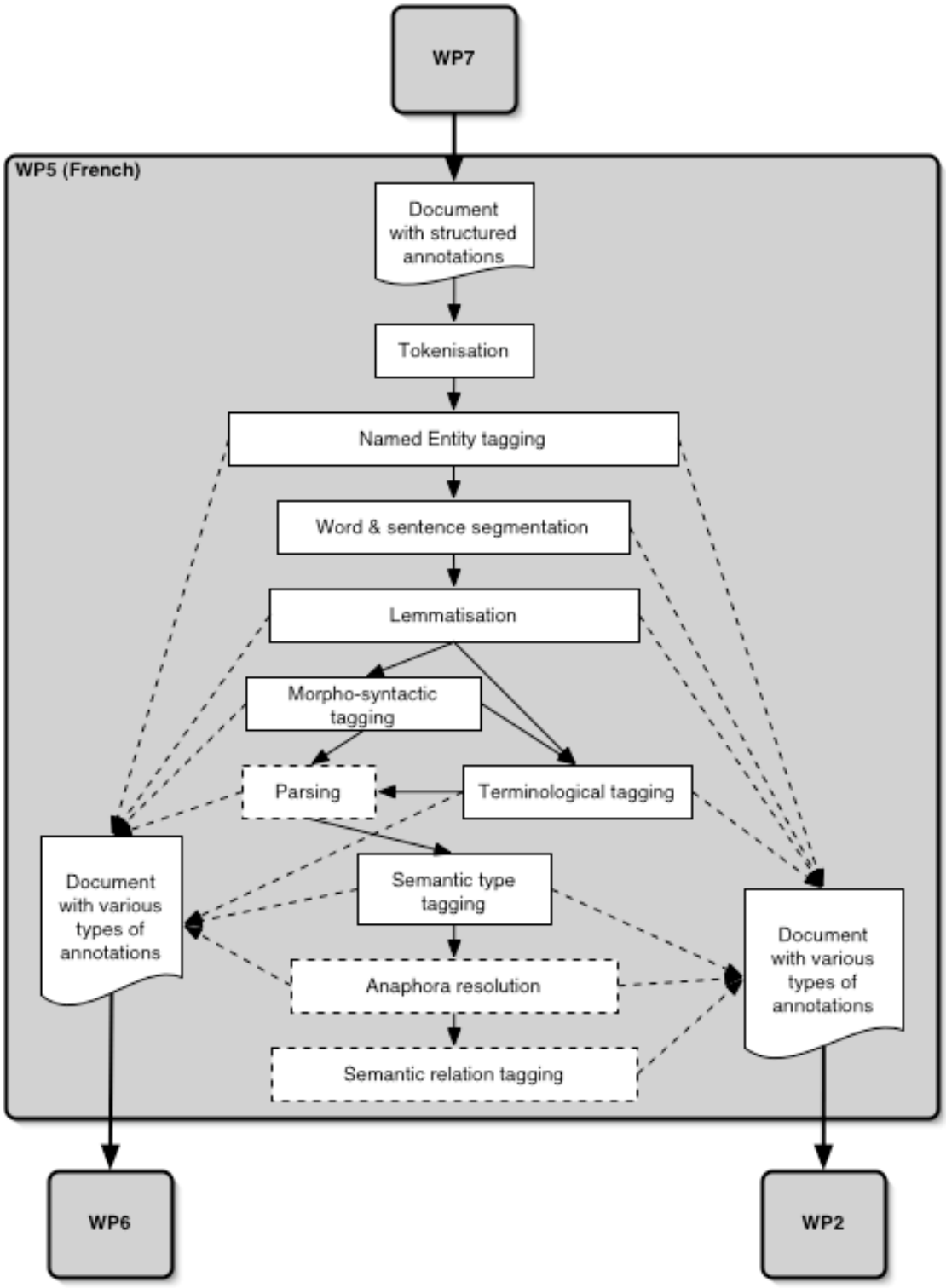


Figure 2 Architecture of the WP5 NL processing line for French.

The NLP line for French is very similar to that for English but the decision of integrating parsing, anaphora resolution and semantic relation tagging will depend on results obtained for English.

5.3 Slovene NL processing line

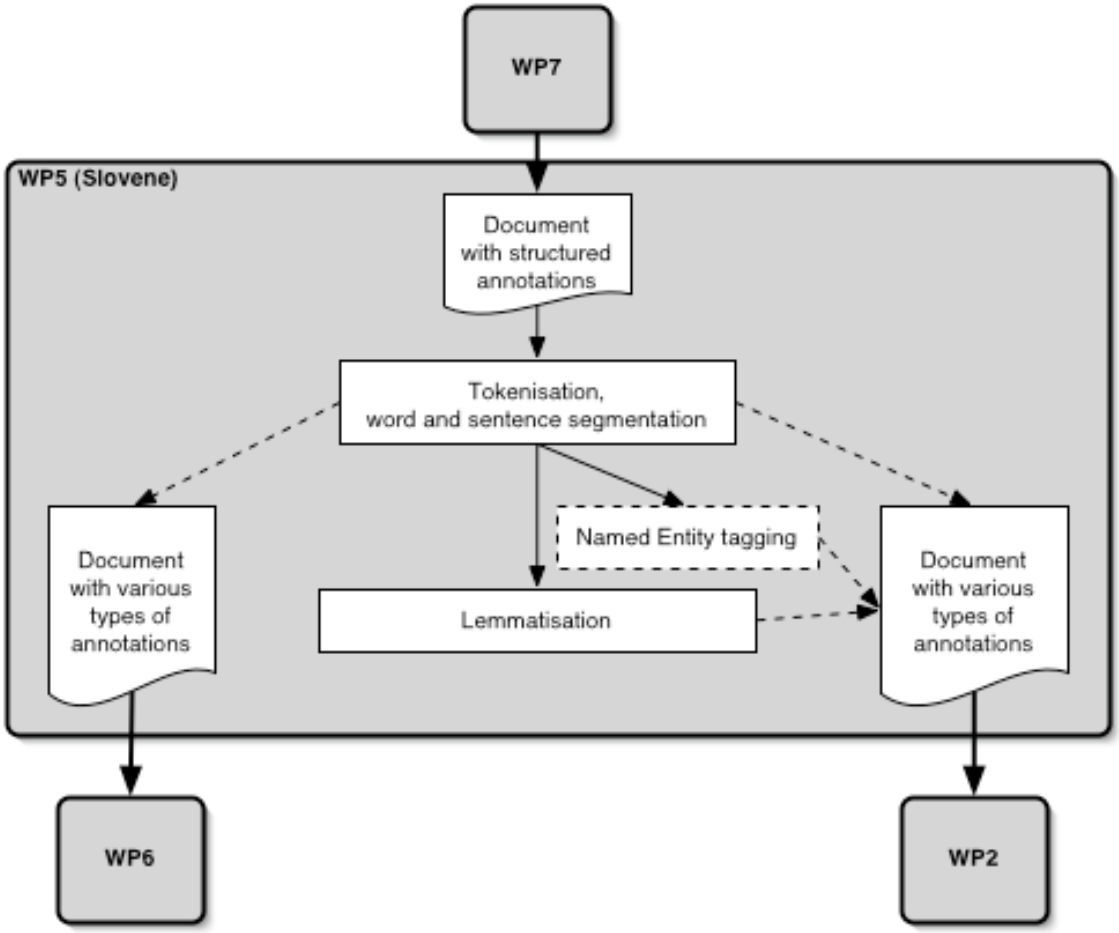


Figure 3 Architecture of the WP5 NL processing line for Slovene.

The Slovene line will have the same structure as the English line, but includes only tokenization, word and sentence segmentation and lemmatisation. Named entity tagging could eventually be added. In the present project, tokenization and word/sentence segmentation are implemented in one block as a pre-processing to lemmatization (no separate modules for word/sentence segmentation). However, we will have to discuss how the English tokeniser and word/sentence segmentation code could be adapted for Slovene.

5.4 Chinese NL processing line

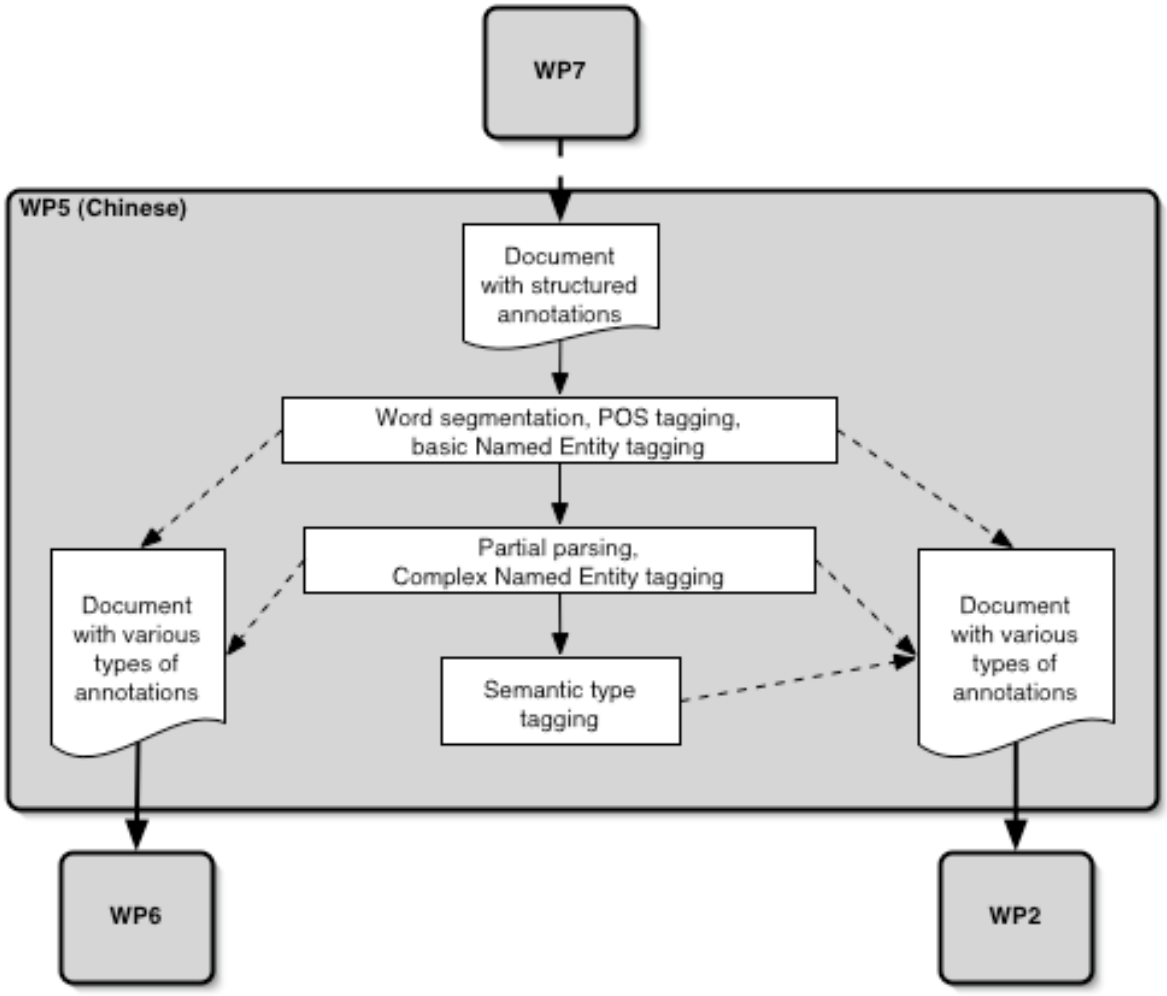


Figure 4 Architecture of the WP5 NL processing line for Chinese.

Because there are not any blanks between different words in Chinese texts, there is no need to add a tokenization stage for Chinese processing. Three main stages are distinguished for Chinese NLP in Alvis project:

- First stage: Chinese segmentation and POS tagging, + basic name entity identification, including personal name and location name.
- Second stage: Chunking, + complex name entity identification, including organization expressions
- Third stage: Give some main semantic types for the identified words and chunks, such as Human, Organization, Location, Time, Numeral, Activity, Attribute, and so on. This information may be topic specific or domain dependent.

These three stages form an information-incremental procedure. Each stage can give useful information for document annotation and indexing strategies.

6 Modules descriptions

This section describes the general specification of the various modules of the NLP line that produces the various types of annotations presented above and that are represented in the architecture diagrams.

6.1 Tokenisation (English, French, eventually Slovene)

6.1.1 Functional description

Tokenisation is the very first step in the linguistic text processing line in WP5. This process ensures that a basic segmentation is done before segmentation can begin. The main purpose is to build tokens as well-identified substrings of the document character string, based upon their types (alphabetic, numeric, symbolic, space characters).

6.1.2 Example

The character string “Spo0A” will be split into 3 different character substrings: “Spo” (alphabetic), “0” (numeric”) and “A” (alphabetic).

6.1.3 Input/output

a. Document input

The input is a collection of Alvis XML documents (in the format described in task 3.2).

b. Augmented document (output)

The output is the original collection in which the canonical document string is segmented into tokens as described in section 4.2.1 c. .

6.2 Name Entity tagging (English, French)

6.2.1 Functional description

The named entity tagger takes as input a tokenized text (see tokenizer section) and produces a text with information about named entities. The named entities are annotated as semantic units, with syntactic and semantic types. Each text sequence corresponding to a named entity will be tagged with a unique tag corresponding to its semantic value (for example a "person" type for person names, "location" type for location names, etc.). All these text sequences are also assumed to be equivalent to nouns: the tagger dynamically produces linguistic units equivalent to words or noun phrases.

The Named entity tagger is based on a set of linguistic resources and grammars as provided by WP6 and described in deliverable D6.2 (see a brief description below).

In Alvis, a rule-based named entity tagger will be provided for French and English. The tool is provided with general domain resources (dictionaries and grammars for person name recognition, location name, dates and other numerical expressions). Specific tagging mode will be possible with additional resources for new domains (e.g. carnivore plants).

6.2.2 Examples

See section 4.4.1 b. for simplified examples of named entity tagging results.

6.2.3 Input/output

a. Document input

The document input in the Alvis format is a tokenized text coming from the Alvis tokenizer module.

b. External resources

The module is based on a set of linguistic resources:

- Lists of trigger words -- *e.g.* `Mr` for `Mister` or `inc.` for `incorporated`

- Gazetteers: large dictionaries of known proper names
- Dictionaries of the general language, essentially to identify unknown words.

Such lists are available on the Internet, for example Consortium for Lexical Research: New Mexico State University (NMSU, http://crl.nmsu.edu), especially CRL-Word lists and Proper names wordlist.

c. Augmented document (output)

The document output is a tokenized text with XML tags corresponding to recognized named entities, in the Alvis format (see above). Each text sequence corresponding to a named entity will be tagged with a unique tag corresponding to its semantic value.

The named entity tagger therefore has an impact on the following annotation levels:

- Words, phrases and semantic units,
- Morpho-syntactic features
- Semantic types

6.2.4 Pending question

The method proposed here is based on gazetteers and dictionaries. Some alternative methods (especially machine learning based tagging, which is different from named entity dictionary acquisition) have been tested in the literature. Whether they could be integrated in the Alvis NLP line remains a pending question that will be discussed.

6.3 Segmentation (English, French, eventually Slovene)

6.3.1 Functional description

The goal of segmentation is to segment the corpus in words and sentences. These very units are the core data required for further linguistic analysis.

Named Entity tagging is performed prior to segmentation so as to resolve some ambiguities. For instance, the string "B. subtilis" could create a problem, given that it contains a period "." that does not mark the end of a sentence. Named entity tagging provides useful information telling us that "B.", is a short form for "Bacillus". Based on such information, word and sentence segmentation is made much easier, because some ambiguities have already been resolved at this stage.

Remaining unresolved cases are processed through simple regular expressions, based on Gregory Grefenstette's work on word segmentation (see•[2]]).

6.3.2 Examples

See sections 4.2.2 and 4.2.3 for examples from different domains and different languages.

6.3.3 Input/output

a. Document input

The input document is provided in the Alvis format. The token annotation level is required. Named entities should also be identified to obtain a better precision at this point.

b. External resources

The script used to perform the segmentation may need a small dictionary to perform pattern matching (i.e. "Dr.", "Mrs.", etc.).

c. Augmented document (output)

The output document contains two extra annotation levels: words and sentences. Words contain references to tokens, and sentences contain references to words.

6.4 Stemming (English, French)

6.4.1 Functional description

The module suppresses the empty words (words that do not have a significant meaning such as grammatical or a-thematic ones) and associates a stem to each remaining word of a sentence. It will be based on a known stemming algorithm such as •[7] .

6.4.2 Examples

See also examples in section 4.3.2

Example of stemming for English in the biomedical domain

Original sentence: *The promoter region of each of these genes has two GerE binding sites.*
Stemmed sentence: *promot region gene gere bind site*

Example of stemming for French in the medical domain

Original sentence: *Toutes ces méthodes permettaient une opacification des artères coronariennes.*
Stemmed sentence: *méthod permet opacif arter*

6.4.3 Input/output

a. Document input

The input document is provided in the ALVIS format. The word level is required.

b. External resources

An external resource should not be necessary. The Porter stemming algorithm is very likely to be used, and it only requires a simple set of rules to perform the stemming of a word.

c. Augmented document (output)

The output document is the input document augmented with the stem annotation level for non empty words.

6.5 Morpho-syntactic tagging (English, French)

6.5.1 Functional description

This module aims at determining the morpho-syntactic category, i.e. the part of speech (POS) tag, of each word. It associates tags to the words that define discourse part and morphologic features.

6.5.2 Examples

The following examples correspond to the TreeTagger (tagset and markup) versions of the examples of section 4.3.3.

Example of tagging for English in the biological domain

This example in English is issued from a biological scientific article abstract downloaded from Medline.

```
<POS cat="IN">During</POS> <POS cat="NN">sporulation</POS> <POS cat="IN">of</POS> <POS
cat="NN">Bacillus</POS> <POS cat="NNS">subtilis</POS> <POS cat=",">,</POS> <POS
cat="NN">spore</POS> <POS cat="NN">coat</POS> <POS cat="NNS">proteins</POS> <POS
cat="VBN">encoded</POS> <POS cat="IN">by</POS> <POS cat="NN">cot</POS> <POS
cat="NNS">genes</POS> <POS cat="VBP">are</POS> <POS cat="VBN">expressed</POS> <POS
cat="IN">in</POS> <POS cat="DT">the</POS> <POS cat="NN">mother</POS> <POS
cat="NN">cell</POS> <POS cat="CC">and</POS> <POS cat="VBD">deposited</POS> <POS
cat="IN">on</POS> <POS cat="DT">the</POS> <POS cat="NN">forespore</POS> <POS
cat="SENT">.</POS>
```

Example of tagging for French in the medical domain

This example in French is issued from a medical manual about coronary diseases. The TreeTagger French version has been used to assign POS tags.

```
<POS cat="POR:IND">Toutes</POS> <POS cat="PRO:DEM">ces</POS> <POS
cat="NOM">méthodes</POS> <POS cat="VER:impf">permettaient</POS> <POS
cat="DET:ART">une</POS> <POS cat="NOM">opacification</POS> <POS cat="PRP:det">des</POS>
<POS cat="NOM">artères</POS> <POS cat="ADJ">coronariennes</POS> <POS cat="SENT">.</POS>
```

6.5.3 Input/output

a. Document input

The input document is provided in the ALVIS format. The required levels are the sentence and the word levels. Named entities could also be identified and, in that case, the semantic_unit level should also be loaded.

b. External resources

No external resource is provided to the module.

c. Augmented document (output)

The module output is the input document augmented with the POS tags at the morphosyntactic_feature level.

6.6 Lemmatisation (English, French, Slovene)

6.6.1 Functional description

The module associates its lemma to each word of a sentence. If the word cannot be lemmatized (for instance a number or a foreign word where none of the rules can be applied), the "lemma tag" will be omitted. For instance, in the Slovene example below, there is no lemma for "24.", "2004", "14.", "3.", "2005".

6.6.2 Examples

See also examples in section 4.3.1.

Example of lemmatisation for English in the biomedical domain

Original sentence: The promoter region of each of these genes has two GerE binding sites.
Lemmatized sentence: the promoter region of each of these gene have two Gere binding site.

Example of lemmatisation for French in the medical domain

Original sentence: Toutes ces méthodes permettaient une opacification des artères coronariennes.
Lemmatized sentence: tout ce méthode permettre un opacification du artère coronarien.

Example of lemmatisation for Slovene

Original sentence: Pocitnice na MPS bodo potekale od 24. decembra 2004 po 14. uri, do 3. januarja 2005".
Lemmatized sentence: Pocitnica na MPS biti potekati od December po ura do januar

6.6.3 Input/output

a. Document input

The input document is provided in the ALVIS format. The word level of annotation is required. Named entities could also be identified. The module may also require morpho-syntactic information in order to get better results. In these cases, the semantic_unit level and the morphosyntactic_features level may be used.

b. External resources

An external resource could be required depending on the choice of the lemmatizer. This resource would provide association between the inflectional forms of a word and its lemma.

c. Augmented document (output)

The module output is the input document augmented with the lemma annotation level.

6.7 Parsing (English)

The parser represents the sentence as a list of words and phrases (constituency syntactic analysis) and/or as a set of syntactic relations between the words (dependency syntactic analysis).

6.7.1 Functional description

The parser finds all the possible parses for the sentence and displays the entire set of solutions.

a. Special features of the parser

The specific problems that the parser has to handle are the following:

- **Coordination**: It is important that the parser resolves the coordination problems.
- **Unknown words**: the parsing must not fail because of unknown words so we need a robust parser.
- **Multi-words expressions**: such linguistic units like terms must be seen as "black boxes" by the parser. This feature prevents the parser from analysing their internal composition (which is already given by the terminological tagging) and, as a result, reduces the parsing complexity and processing time.

b. Expected quality of the parsing results

Due to the language complexity, one cannot hope to get a perfect and univocal parsing of an entire corpus. Perfect parsing can occur only on rather simple/short sentences or on a controlled language (in technical domains). The grammar adaptation (WP6) and a good corpus normalisation must lead to a "as best as possible" parse in focusing on particular parts of the sentence that will produce relations relevant to the further processing steps (in WP5 and WP6).

Erroneous parsing(s)

Local parsing errors (affecting a relation between two units) are generally due to lacks in the lexicon that induce a wrong category assignation (based on morphological features). The use of the morpho-syntactic tagging would reduce their number, as explained in section 6.8.

Global parsing errors (affecting the entire structure of the sentence) mainly occur on long distance dependencies when several propositions are coordinated or embedded. These errors can be reduced by a good adaptation/construction of the grammar resulting from a deep analysis of the characteristics of the language concerned, possibly domain-specific.

Incomplete parsing(s)

The parse can fail to find at least one complete parse of the sentence. This is not rare, especially when dealing with specific domain texts. This is due to two main reasons:

- the dictionary-grammar is not complete: one or more link between words is missing. Technical texts are very tricky since sentences contain formulas, numbers, abbreviations, etc. A hard work of adaptation must be done on the dictionary-grammar to fix these problems.
- the sentence is linguistically incorrect. This case is rather common in the biological texts we used to test the parser and which are sometimes written by non-native English speakers. One solution is to relax constraints in the grammar. For instance, compulsory determiners on some nouns have been made optional.

Evaluation should focus on the specific relation types that are pertinent to the semantic tagging or the ontology learning.

- In extreme cases, a complex sentence could remain unparsed, due to a parsing failure or an excessive processing time.

Multiple parses

In case of ambiguous sentences (eventually erroneous ambiguous sentences), the parser can choice one parse or produces several parses (like with the Link Parser). In this latter case, it is important to have a good ordering procedure based on the quality of the different analyses

Processing time

This is the critical point of the syntactic parsing since the quality of the results is dependent on the processing time. The more the parsing is precise and informative, the more it is time consuming. This processing time may be prohibitive for documents addressed to WP2 (production NLP line), where the volume of document to process is important. For the acquisition phase, which concerns smaller corpora, a higher parsing time is acceptable (a couple of days seems to be a limit for an acquisition corpus).

We are convinced that a good recognition of the terms can reduce significantly the number of possible parses and as a result the processing time for syntax. This conviction has guide the conception of the WP5 architecture between parsing and terminological tagging (see Figure 1).

6.7.2 Examples

See section 4.3.4

6.7.3 Input/output

a. Document input

The word level of annotation is required. Depending on the choice of the parser, the morphosyntactic level may be needed. Term identification facilitates the parsing.

Even when the parser performs its own morpho-syntactic tagging, it must take into consideration pre-existing morpho-syntactic ones. This avoids the augmentation of the parser's lexicon with the terms and named entities that have been identified before in the processing line. This is also important for modularity, since most syntactic parsers work on pre-tagged corpora.

b. External resources

The “Link Grammar Parser”, a “dependency-based parser” which aims at finding syntactic relations between words, currently performs the syntactic parsing task. The formalism is quite different from the classical (constituency-based)

parsers' grammars, which are made of syntactic rules. The Link parser's grammar also contains rules but each of these rules holds for a set of words that are listed after it. In this way, the grammar not only contains the grammar rules but also the lexicon.

```
synthesizes, products.v : Ssg- O+
system, gene : AN- Dt- O- ( Ssg+ or AN+ )
products.n : AN- Dt- O- ( Spl+ or AN+ )
the: Dt+
```

Figure 5 Except from the Link Grammar

The rules express the possible links that connect a word to the others, on its left (-) and on its right (+). In the Figure 5, the words "synthesizes" and "products" (verbs) can have a singular subject link on their left and an object link on their right. Note that "products" can also be analysed as a plural noun. Another grammar entry is used to declare its syntactic behaviour in that case.

More details about the grammar format are given in Document D6.2 about the "Requirements for integration of WP6 results into WP5 normalization and representation tasks and into WP9 query refinement task".

c. Augmented document (output)

The output is the annotated document augmented with the syntactic_relation annotation level.

6.8 Terminology tagging (English, French)

6.8.1 Functional description

This module aims at recognizing terms in the documents. Term lists are provided by the WP6 package. Term variants of a main term can be associated with the terms in the lists. In that case, the main term can be associated to the term found in the "canonical_form" element at the semantic_unit level. Depending on the quality of the terminological resources it is based on, the terminological tagger may also associates some semantic categories to the terms.

6.8.2 Examples

See section 4.2.4 b. for simplified examples of term tagging results on different domains and different languages.

6.8.3 Input/output

a. Document input

Two levels of annotations are required: the word level and morpho-syntactic features. Existing phrases can also be taken into account.

b. External resources

Some external resources are required. They must be adapted to the domain of the document. For instance, term tagging a Medline biomedical abstract requires terms list learned on a biomedical document corpus. Such lists are built by the WP6 (see deliverable D6.2).

c. Augmented document (output)

When a term is found in the text, a unit is created at the semantic_unit level. This unit can refer either to a word or a phrase. A semantic type may be associated with it.

6.9 Semantic type tagging (English, French, eventually Chinese)

As presented in section 3, the ontology defines two types of knowledge:

- Concepts organized in generality hierarchies, where the leaves are domain specific terms,
- Domain specific relations between those terms.

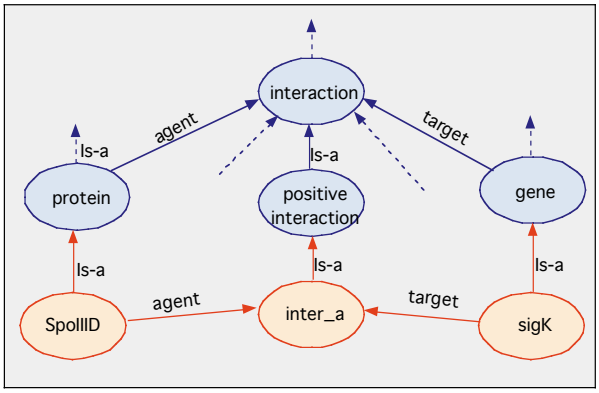


Figure 6 Extract of an ontology in the genomics domain

WP5 semantic type tagging exploits this resource to annotate pre-processed documents. It tags the relevant semantic units (usually named-entities and terms) by concepts according to the ontology. For instance, *SpoIIID* is tagged by *Protein* and by *biological object* in the sentence "SpoIIID stimulates sigK transcription" according to the ontology of Figure 6. Furthermore, the module may declare a word as a semantic unit and tag it accordingly to the ontology, like “stimulates” in this example.

6.9.1 Input/output

a. Document input

Semantic tagging is done on the results provided by the syntactic parsing module, or the term recognition module, depending on the processing mode.

b. External resources

The external ontology is provided by WP6. The syntactic constraints if available will be exploited depending on the mode.

c. Augmented document (output)

The document output is augmented with the semantic_features level, which contains semantic information about the semantic units (either identified at the lower levels or added during the semantic tagging) according to the Alvis format described in section 4.4.1.

6.9.2 Pending questions

As described above, depending on the mode, the semantic typing will rely or not on syntactic parsing. In any of these cases, there are two main issues to discuss:

- In case of multiple hierarchies, what path(s) should be chosen for the annotation?
- What is the appropriate conceptual level for tagging the document?

c. Semantic ambiguity resolution

An ontology may contain multiple hierarchies, which leads to ambiguous tagging. The word *cat*, for instance, can be annotated by multiple paths. The selection of the appropriate one obviously depends on the meaning of the text *cat* occurs in.

Semantic typing of the word "cat" requires some semantic disambiguation for choosing among the 5 mutually exclusive solutions.

1. catalase -> enzyme -> protein
2. chloramphenicol acetyltransferase ->transferase->enzyme->protein
3. mammalian -> animal
4. chlamydia antibody testing -> antibody screening technique-> experimental method
5. gene -> DNA sequence

In Alvis, the selection of the appropriate path is done with the help of syntactic constraints (represented in red in the following table), in the form of selectional restrictions, when available and if the document collection is syntactically parsed.

Noun : test Adj: chlamydia antibody testing	Noun : disease NprepN(of/in) : mammalian	Noun : expression Adj : gene

In the other case, the tag is chosen on the basis of the neighbouring words of the semantic unit to be tagged and not of a syntactic context in the document. If there are still multiple solutions, then the document is annotated with all solutions, ordered by plausibility order.

d. Generality level

Each semantic unit referred in the ontology can be tagged by all the ascendants to the root(s) of the ontology. However, all levels are not suitable in the IR framework. For instance, in a collection of scientific papers, a paper on Machine Learning should be considered as more similar to a paper on Artificial Intelligence, than a paper on Physics.

The appropriate generality level obviously depends on the indexing scheme. Three different options are still discussed. We will briefly present them here, but they have no main effect on the annotation format.

The selection of the suitable generality level can be done during the ontology building by WP6, by semantic typing by WP5 NLP line, or postponed to the probabilistic model computation by WP2.

1. WP6 ontology building tool could label the ontology concepts with relevancy weight to be computed from training corpora or given by a domain expert. Then WP5 NLP line or WP2 probabilistic model would choose the suitable conceptual level according to this score information. This option is preferable in the case of the global knowledge of the ontology is required for scoring the nodes, for instance in the case where the granularity varies a lot within the ontology. For instance, very general nodes are less relevant in a fine-grained ontology.
2. WP5 NLP annotation process could also more or less arbitrarily choose one conceptual level, such as the first ascendant. This process is the one that has the most local view as opposed to WP6 ontology design and WP2 probabilistic model computation.
3. WP2 probabilistic model component could select the appropriate description level for each term in the document, according the possible annotations done by WP5 and its knowledge of the whole document collection.

6.10 Anaphora resolution (English)

6.10.1 Functional description

The Anaphora resolution module takes as input an annotated text and produces a text with information about anaphoric relations. The anaphoric resolution module is based on the resources provided either by previous WP5 modules or by external linguistic resources and a probabilistic model, Bayesian network as provided by WP6. The main advantages of this combined approach is to:

- Merge all the resource into a unique formalism
- Be easily adapted to new domains (by the way of specific domain probabilistic parameters)

The module considers both English and French languages. It will only handle pronominal anaphora.

6.10.2 Examples

See examples in section 4.4.2 b.

6.10.3 Input/Output

a. Document input

The anaphora resolution module takes as input an annotated document coming from the Semantic type tagging module, in the Alvis format.

b. External resources

The anaphora resolution module relies on an XML file which represent the domain specific parameters for the Bayesian network, as describe in deliverable D6.2 (section 6.3: *Resources for anaphora resolution*). This resource is provided by WP6.

The module needs external linguistic resources. These resources are the same as Named entities recognition module's linguistic resources, i.e. dictionaries and grammars encoded through finite state transducers (for more details see D6.2).

c. Augmented document (output)

The document output is an augmented text with XML tags corresponding to anaphora relations between antecedents and pronouns, in the Alvis format (see above). Since, a new relation allows to make inferences about the anaphora semantic type, the module has also an impact on the semantic type annotation level.

6.11 Domain specific semantic relation tagging (English)

6.11.1 Functional description

Domain specific semantic relations identify relations occurring between *instances* of the ontological concepts in the document. These instances are identified and tagged accordingly by the semantic typing module. As a result, these semantic relation annotations give another level of semantic representation of the document that makes explicit the role of semantic units (usually named-entities and/or terms, see section 4.2.4) play with respect to each other pertaining to the ontology of the domain.

However, this annotation depends on previous document annotations and we have to consider two different tagging strategies, depending on the two different processing strategies described in sections 2.4and 3.4:

- If the document collection is parsed, the module can exploit this information to tag relations mentioned explicitly. This is achieved through the pattern matching of information extraction rules provided by WP6 (D6.2, section 7.3). It is the responsibility of WP6 to provide, besides the information extraction rules, the tagging module that exploits them.
- In the case where the document is not syntactically parsed, the module will base its tagging on relations given by the ontology provided by WP6, that is to say all known relations holding between semantic units described in the document will be added, whether those relations be explicitly mentioned in the document or not.

As noted earlier, this choice will impact the IR task. We discuss related issues in section 6.11.3.

6.11.2 Input/output

a. Document input

The semantic relation tagging is built upon the results provided by the semantic type module (section 6.9). Therefore, it takes as input a document with semantic units annotated (the *semantic_unit_level* and *semantic_feature_level* XML blocks).

b. External resources

The external resources depend on the final processing line:

- If the document is not syntactically parsed, the module uses the same ontology as the semantic type tagger, provided by WP6.
- If the document is syntactically parsed, the module needs a set of IE rules, the pattern matcher that exploits them and also the ontology used by the semantic type module. The ontology is needed to recover the type of the ontology node IDs to perform the pattern matching of the IE rule.

c. Augmented document (output)

The document output is the partially annotated input document, augmented with some domain specific semantic relations according to the Alvis format described in section 4.4.2.

6.11.3 Pending questions

Two strategies have been proposed for semantic relation tagging and it is WP2 and WP8 responsibility to figure out which one to use to index the documents. However, strategy 1 may not be technically satisfactory from the efficiency and quality point of view. First, tagging explicit relations requires to parse the entire document that is very time-consuming and might not be valid at the WP5 level. Second, it is expected that the quality of indexing performed by WP2 is correlated to the quality of the semantic relation extraction. This latter quality does not depend on WP5 but on the set of IE rules acquired by WP6. The assessment of performance they will provide may dictate the choice of the semantic relation tagging strategy, independently of WP2 results.

6.12 Word segmentation, POS tagging and basic name entity identification (Chinese)

6.12.1 Functional description

This module is an integrated part for Chinese word segmentation, Part-of-Speech(POS) tagging, basic proper names identification (including Chinese personal names, Chinese location names and the foreign translation names). Here, different kinds of language resources, such as Chinese lexicons with word and POS information, large-scale Chinese corpus annotated with correct word segmentation and POS tag information, special knowledge base for Chinese proper name identification, are very useful for Segmentation and Tag processing and disambiguation.

6.12.2 Example

See example in section 4.5.

6.12.3 Input/output

a. Document input

The input is a collection of Alvis XML documents (in the format described in task 3.2).

b. External resources

Different kinds of language resources, such as Chinese lexicons with word and POS information, large-scale Chinese corpus annotated with correct word segmentation and POS tag information, special knowledge base for Chinese proper name identification, are very useful for Segmentation and Tag processing and disambiguation.

c. Augmented document (output)

The document output is the input document enriched with annotations at the following levels: word, morpho-syntactic features, semantic units and semantic types.

6.13 Partial parser (Chinese)

6.13.1 Functional description

This module will focus on the identification of basic chunks (simple phrases), including basic noun phrases and basic verb phrases in the Chinese sentences. Here, the key point is to determine the correct boundaries of some complex noun chunks, such as complex time, location and organization expressions.

6.13.2 Example

See example in section 4.5.

6.13.3 Input/output

a. Document input

The input is a document as delivered as output from the Segmentation and Tag module described above (section 6.12).

b. Augmented document (output)

The document output is the input document augmented with phrase annotations corresponding to the chunking result of the sentences.

6.14 Semantic type tagger (Chinese)

6.14.1 Functional description

This module will give some main semantic types for the identified words and chunks. These types include: Human, Organization, Location, Time, Numeral, Activity, Attribute, and so on. This information may be topic specific or domain dependent. Most of this information can be extracted from a large scale Chinese semantic dictionary – HowNet (http://www.keenage.com).

6.14.2 Example

See example in section 4.5.

6.14.3 Input/output

a. Document input

The input is a document as delivered as output from the Partial parsing module described above (section 6.13).

b. External resource

The large scale Chinese semantic dictionary – HowNet.

c. Augmented document (output)

The output is the input document augmented with semantic type information attached to each content words and chunks.

7 Conclusions

The present deliverable has stated the specifications for the development of the WP5 linguistic processing line for four languages, with different levels of achievement expected in each language. WP5 will be able to deliver on demand

various types of linguistically enriched documents to WP6 (for acquisition purposes) or to WP2 (for indexing and IR purposes).

Further work in WP5 will focus on the following points:

- Development and integration of the individual modules in the various languages;
- Test on the performances and optimization for the analysis of large amount of textual data (some of the collections contain 20 Gigabytes of documents.
- Definition of the experimental protocol for the integration of NLP in IR: this implies to define the various levels of annotated documents that should be used in IR experiments and that WP5 will therefore actually have to produce. Only a subset of the annotated documents that WP5 may output (see the doted arrows of the diagrams in section 5) will be tested in IR experiments, and it must be chosen after discussions among Alvis partners, on the basis of
  - the state of the art in that domain,
  - the expected quality of WP5 outputs,
  - the intrinsic properties of Alvis search engine modules (notably, the probabilistic model).

8 References

[1] Bird S. and Liberman M. “Annotation graphs as a framework for multidimensional linguistic data analysis”. In NJ: Association for Computational Linguistics, editor, *Towards Standards and Tools for Discourse Tagging - Proceedings of the Workshop*, pages 1–10, Somerset, 1999.

[2] Grefenstette G. *Explorations in Automatic Thesaurus Discovery*. Kluwer Academic Press, Boston, 1994.

[3] Dzeroski S., Erjavec T. “Machine Learning of Mophosyntactic Structure: Lemmatising Unknown Slovene Words”. *Applied Artificial Intelligence*, 2004.

[4] Grishman R.. “Tipster architecture design document version 2.3”. *Technical report*, DARPA, 1997.

[5] Krovetz R.: "Viewing morphology as an inference process," in R. Korfhage et al., Proc. 16th ACM SIGIR Conference, Pittsburgh, June 27-July 1, 1993; pp. 191-202.

[6] Mladenic D. “Learning Word Normalization Using Word Suffix and Context from Unlabeled Data”. *Proc. ICML*, 2002.

[7] Porter M. “An algorithm for Suffix Stripping”. *Proc. ACM SIGIR Conference on Research and Development in Information Retrieval*, 1980.

9 Annex: complete DTD for linguistic annotations

```
<?xml version="1.0" encoding="UTF-8"?>

<!-- ============================================================ -->
<!--                    DTD definition - ALVIS WP5                -->
<!-- ============================================================ -->

<!--                    linguistic annotation (ALVIS WP5)         -->

<!ELEMENT  linguisticAnalysis
                        (token_level,
                         word_level,
                         sentence_level,
                         phrase_level,
                         semantic_unit_level,
                         lemma_level,
                         stem_level,
                         morphosyntactic_features_level,
```

```
                        syntactic_features_level,
                        syntactic_relation_level,
                        semantic_features_level,
                        semantic_relation_level,
                        anaphora_relation_level                )    >

<!-- ========================================================== -->
<!--                    TOKEN LEVEL                             -->
<!-- ========================================================== -->
<!ELEMENT  token_level  (token_sep | token_alpha | token_symb |
                         token_num )+                             >

<!--                    separator token                         -->
<!ELEMENT  token_sep    (id, content, from, to)                   >

<!--                    alphanumeric token                      -->
<!ELEMENT  token_alpha  (id, content, from, to)                   >

<!--                    symbol token                            -->
<!ELEMENT  token_symb   (id, content, from, to)                   >

<!--                    numeric token                           -->
<!ELEMENT  token_num    (id, content, from, to)                   >

<!--                    content of the token                    -->
<!ELEMENT  content      (#PCDATA)                                 >

<!--                    start offset of the token               -->
<!ELEMENT  from         (#PCDATA)                                 >

<!--                    end offset of the token                 -->
<!ELEMENT  to           (#PCDATA)                                 >



<!-- ========================================================== -->
<!--                    WORD LEVEL                              -->
<!-- ========================================================== -->
<!ELEMENT  word_level   (word)+                                   >

<!--                    word                                    -->
<!ELEMENT  word         (id, list_refid_token, form*)             >

<!--                    id of the element                       -->
<!ELEMENT  id           (#PCDATA)                                 >

<!--                    list of the tokens which compose the words -->
<!ELEMENT  list_refid_token
                        (refid_token)+                           >

<!--                    token id, part of the words             -->
<!ELEMENT  refid_token  (#PCDATA)                                 >

<!--                         form                               -->
<!ELEMENT  form    (#PCDATA)                                      >

<!-- ========================================================== -->
<!--                    SENTENCE LEVEL                          -->
<!-- ========================================================== -->
<!ELEMENT  sentence_level (sentence)+                             >

<!--                    sentence                                -->
<!ELEMENT  sentence     (id, refid_start_token, refid_end_token,
                         list_refid_word, form*)                  >
```

```
<!--                    Reference of the token at the beginning    -->
<!--                    of the sentence                            -->
<!ELEMENT  refid_start_token
                        (#PCDATA)                                   >

<!--                    Reference of the token at the end          -->
<!--                    of the sentence                            -->
<!ELEMENT  refid_end_token
                        (#PCDATA)                                   >

<!--                    list of the words which compose the word   -->
<!ELEMENT  list_refid_word
                        (refid_word)+                               >

<!--                    word id, part of the word                  -->
<!ELEMENT  refid_word  (#PCDATA)                                    >


<!-- ========================================================== -->
<!--                    PHRASE LEVEL                            -->
<!-- ========================================================== -->
<!ELEMENT  phrase_level
                        (phrase)+                                   >

<!--                    phrase                                     -->
<!ELEMENT  phrase      (id, list_refid_components, form*)           >

<!--                    list_refid_components                      -->
<!ELEMENT  list_refid_components
                        (refid_word | refid_phrase)+                >

<!--                    refid_phrase                              -->
<!ELEMENT  refid_phrase
                        (#PCDATA)                                   >

<!-- ========================================================== -->
<!--                    SEMANTIC UNIT                           -->
<!-- ========================================================== -->
<!--                    Named entities and terms                   -->
<!ELEMENT  semantic_unit
                        (named_entity | term | undefined)+                      >

<!--                    named entity                               -->
<!ELEMENT  named_entity (id, refid_phrase | refid_word, form*,
                         canonical_form*, named_entity_type)          >

<!ELEMENT  list_refid_semantic_unit (refid_semantic_unit)+          >
<!ELEMENT  refid_semantic_unit (#PCDATA)                            >

<!--                    named_entity_type                          -->
<!ELEMENT  named_entity_type
                        (#PCDATA)                                   >

<!--                    list_refid_phrase                         -->
<!ELEMENT  list_refid_phrase
                        (refid_word | refid_phrase)+                >

<!--                    term                                       -->
<!ELEMENT  term         (id, refid_phrase | refid_word, form*,
                         canonical_form*)                           >

<!-- ========================================================== -->
<!--                    LEMMA LEVEL                             -->
```

```
<!-- ============================================================ -->
<!ELEMENT  lemma_level  (lemma)+                                    >

<!--                    lemma                                     -->
<!ELEMENT  lemma        (id, canonical_form+, refid_word,
                         form*)                                    >

<!--                    canonical form of the word                -->
<!--                    corresponding to the lemma                -->
<!ELEMENT  canonical_form
                        (#PCDATA)                                  >

<!-- ============================================================ -->
<!--                    STEM LEVEL                                -->
<!-- ============================================================ -->
<!ELEMENT  stem_level   (stem)+                                    >

<!--                    stem                                      -->
<!ELEMENT  stem         (id, stem_form+, refid_word,
                         form*)                                    >

<!--                    stem form                                 -->
<!ELEMENT  stem_form    (#PCDATA)                                  >

<!-- ============================================================ -->
<!--                    MORPHOSYNTACTIC FEATURES LEVEL              -->
<!-- ============================================================ -->

<!ELEMENT  morphosyntactic_features_level
                        (morphosyntactic_features)+                  >

<!--                    morphosyntactic_features                    -->
<!ELEMENT  morphosyntactic_features
                        (id, refid_word | refid_phrase, tense*, aspect*,
                         person*, number*, voice*, gender*,
                         syntactic_category, form*)            >
<!--                    tense                                     -->
<!ELEMENT  tense        (#PCDATA)                                  >
<!--                    aspect                                    -->
<!ELEMENT  aspect       (#PCDATA)                                  >
<!--                    number                                    -->
<!ELEMENT  number       (#PCDATA)                                  >
<!--                    person                                    -->
<!ELEMENT  person       (#PCDATA)                                  >
<!--                    voice                                     -->
<!ELEMENT  voice        (#PCDATA)                                  >
<!--                    gender                                    -->
<!ELEMENT  gender       (#PCDATA)                                  >
<!--                    syntactic_category                        -->
<!ELEMENT  syntactic_category
                        (#PCDATA)                                  >



<!-- ============================================================ -->
<!--                    SYNTACTIC_RELATION_LEVEL                  -->
<!-- ============================================================ -->
<!ELEMENT  syntactic_relation_level
                        (syntactic_relation)+                      >

<!--                    syntactic_relation                        -->
<!ELEMENT  syntactic_relation
                        (id, syntactic_relation_type,
                         refid_head,
```

```
                        refid_modifier)                                  >

<!--                    refid_head phrase or word                      -->
<!ELEMENT  refid_head
                        (refid_word | refid_phrase)                      >

<!--                    refid_modifier phrase or word                  -->
<!ELEMENT  refid_modifier
                        (refid_word | refid_phrase)                      >


<!--                    syntactic_relation_type                        -->
<!ELEMENT  syntactic_relation_type
                        (#PCDATA)                                        >


<!-- ========================================================== -->
<!--                    SEMANTIC_FEATURES_LEVEL                       -->
<!-- ========================================================== -->
<!ELEMENT  semantic_features_level
                        (semantic_features)+                              >

<!--                    semantic_features                              -->
<!ELEMENT  semantic_features
                        (id, semantic_category, refid_semantic_unit)  >

<!--                    semantic_category                              -->

<!ELEMENT  semantic_category
                        (list_refid_ontology_node)+                      >

<!--                    list_refid_ontology_node                       -->
<!ELEMENT  list_refid_ontology_node
                        (refid_ontology_node)+                           >

<!--                    refid_ontology_node                            -->
<!ELEMENT  refid_ontology_node
                        (#PCDATA)                                        >

<!-- ========================================================== -->
<!--                    DOMAIN_SPECIFIC_RELATION_LEVEL             -->
<!-- ========================================================== -->
<!ELEMENT  domain_specific_relation_level
                        (domain_specific_relation)+                      >

<!--                    domain_specific_relation                   -->
<!ELEMENT  domain_specific_relation
                        (id, domain_specific_relation_type,
                         refid_semantic_unit, refid_semantic_unit)   >

<!--                    domain_specific_relation_type              -->
<!ELEMENT  domain_specific_relation_type
                        (#PCDATA)                                        >


<!-- ========================================================== -->
<!--                    ANAPHORA_RELATION_LEVEL                       -->
<!-- ========================================================== -->
<!ELEMENT  anaphora_relation_level
                        (anaphora_relation)+                             >
<!--                    anaphora_relation                              -->
<!ELEMENT  anaphora_relation
                        (id, anaphora_relation_type,
                         anaphora, antecedent)                           >
```

```
<!ELEMENT  antecedent
                        (list_refid_semantic_unit)+                  >
<!ELEMENT  anaphora
                        (refid_semantic_unit)                        >


<!--                    anaphora_relation_type                      -->
<!ELEMENT  anaphora_relation_type
                        (#PCDATA)                                    >
```